TUGAS AKHIR – TE 145561

KONTROL KECEPATAN MOTOR INDUKSI 3 FASA 1/4 PK DENGAN MIKROKONTROLER DAN DIMONITOR MENGGUNAKAN KOMPUTER DENGAN SERIAL KOMUNIKASI *ETHERNET*

Ika Hasfritasari
NRP. 2213 039 002
Cahyo Aditya Laksono
NRP. 2213 039 031

Dosen Pembimbing
Ir. Josaphat Pramudijanto, M.Eng
Agus Suhanto, S.Pd

PROGRAM STUDI D3 TEKNIK ELEKTRO
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember
Surabaya 2016

FINAL PROJECT – **TE 145561**

INDUCTION MOTOR SPEED CONTROL 3 PHASE 1/4 PK WITH MICROCONTROLLER AND MONITORED BY COMPUTER USING ETHERNET SERIAL COMMUNICATION

Ika Hasfritasari
NOR 2213 039 002
Cahyo Aditya Laksono
NOR 2213 039 031

Supervisor
Ir. Josaphat Pramudijanto, M.Eng
Agus Suhanto, S.Pd

ELECTRICAL ENGINEERING D3 STUDY PROGRAM
Faculty of Industrial Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2016

LEMBAR PERNYATAAN
PERSETUJUAN PUBLIKASI KARYA ILMIAH
UNTUK KEPENTINGAN AKADEMIS

Sebagai mahasiswa Institut Teknologi Sepuluh Nopember Surabaya, yang bertanda tangan di bawah ini saya :

Nama : Ika Hasfritasari

Nrp. : 2213 039 002

Jurusan / Fak. : D3 Teknik Elektro / FTI

Alamat kontak : Jl Demang Sari, RT 02/ RW 01, Desa Keboan Anom, Gedangan, Sidoarjo

a. Email : hasfritasari.ika@gmail.com

b. Telp/HP : 087855621929

Menyatakan bahwa semua data yang saya *upload* di Digital Library ITS merupakan hasil final (revisi terakhir) dari karya ilmiah saya yang sudah disahkan oleh dosen penguji. Apabila dikemudian hari ditemukan ada ketidaksesuaian dengan kenyataan, maka saya bersedia menerima sanksi.

Demi perkembangan ilmu pengetahuan, saya menyetujui untuk memberikan **Hak Bebas Royalti Non-Ekslusif** ***(Non-Exclusive Royalti-Free Right)*** kepada Institut Teknologi Sepuluh Nopember Surabaya atas karya ilmiah saya yang berjudul :

Kontrol Kecepatan Motor Induksi 3 Fasa 1/1 PK dengan Mikrokontroler dan Dimonitor Menggunakan Komputer dengan Serial Komunikasi Ethernet.

Dengan Hak Bebas Royalti Non-Ekslusif ini, Institut Teknologi Sepuluh Nopember Surabaya berhak menyimpan, mengalih-media/format-kan, mengelolanya dalam bentuk pangkalan data *(database)*, mendistribusikannya, dan menampilkan/mempublikasikannya di internet atau media lain untuk kepentingan akademis tanpa meminta ijin dari saya selama tetap mencantumkan nama saya sebagai penulis/pencipta. Saya bersedia menanggung secara pribadi, segala bentuk tuntutan hukum yang timbul atas pelanggaran Hak Cipta dalam karya Ilmiah saya ini tanpa melibatkan pihak Institut Teknologi Sepuluh Nopember Surabaya.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Surabaya

Pada tanggal : 28 Juni 2015

Yang menyatakan,

Dosen Pembimbing 1

Ir. Josaphat Pramudijanto, M.Eng

NIP. 19621005 199003 1 003

IKA Hasfritasari

Nrp. 2213 039 002

KETERANGAN :

Tanda tangan pembimbing wajib dibubuhi stempel jurusan.

Form dicetak dan diserahkan di bagian Pengadaan saat mengumpulkan hard copy TA/Tesis/Disertasi.

LEMBAR PERNYATAAN
PERSETUJUAN PUBLIKASI KARYA ILMIAH
UNTUK KEPENTINGAN AKADEMIS

Sebagai mahasiswa Institut Teknologi Sepuluh Nopember Surabaya, yang bertanda tangan di bawah ini saya :

Nama : Cahyo Aditya Laksono

Nrp. : 2213 039 031

Jurusan / Fak. : D3 Teknik Elektro / FTI

Alamat kontak : Pondok Sidokare Indah W-13

a. Email : cahyoadityaa@gmail.com

b. Telp/HP : 089675833977

Menyatakan bahwa semua data yang saya *upload* di Digital Library ITS merupakan hasil final (revisi terakhir) dari karya ilmiah saya yang sudah disahkan oleh dosen penguji. Apabila dikemudian hari ditemukan ada ketidaksesuaian dengan kenyataan, maka saya bersedia menerima sanksi.

Demi perkembangan ilmu pengetahuan, saya menyetujui untuk memberikan **Hak Bebas Royalti Non-Ekslusif** ***(Non-Exclusive Royalti-Free Right)*** kepada Institut Teknologi Sepuluh Nopember Surabaya atas karya ilmiah saya yang berjudul :

Kontrol Kecepatan Motor Induksi 3 Fasa 1/4 PK dengan Mikrokontroler dan Dimonitor Menggunakan Komputer dengan Serial Komunikasi Ethernet

Dengan Hak Bebas Royalti Non-Ekslusif ini, Institut Teknologi Sepuluh Nopember Surabaya berhak menyimpan, mengalih-media/format-kan, mengelolanya dalam bentuk pangkalan data *(database)*, mendistribusikannya, dan menampilkan/mempublikasikannya di internet atau media lain untuk kepentingan akademis tanpa meminta ijin dari saya selama tetap mencantumkan nama saya sebagai penulis/pencipta. Saya bersedia menanggung secara pribadi, segala bentuk tuntutan hukum yang timbul atas pelanggaran Hak Cipta dalam karya Ilmiah saya ini tanpa melibatkan pihak Institut Teknologi Sepuluh Nopember Surabaya.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Surabaya

Pada tanggal : 28 Juni 2015

Yang menyatakan,

Dosen Pembimbing 1

Ir. Josaphat Pramudijanto, M.Eng

NIP. 19621005 199003 1003

Cahyo Aditya Laksono

Nrp. 2213 039 031

KETERANGAN :

Tanda tangan pembimbing wajib dibubuhi stempel jurusan.

Form dicetak dan diserahkan di bagian Pengadaan saat mengumpulkan hard copy TA/Tesis/Disertasi.

KONTROL KECEPATAN MOTOR INDUKSI 3 FASA 1/4 PK DENGAN MIKROKONTROLER DAN DI*MONITOR* MENGGUNAKAN KOMPUTER DENGAN SERIAL KOMUNIKASI *ETHERNET*

TUGAS AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan
Untuk Memperoleh Gelar Ahli Madya Teknik
Pada
Bidang Studi Elektro Industri
Program Studi D3 Teknik Elektro
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember

Menyetujui :

Dosen Pembimbing I	Dosen Pembimbing II
Ir.Josaphat Pramudijanto, M.Eng	Agus Suhanto, S.Pd
NIP. 19621005 199003 1 003	NIP. 19650821 198603 1 010

SURABAYA
JUNI, 2016

KONTROL KECEPATAN MOTOR INDUKSI 3 FASA 1/4 PK DENGAN MIKROKONTROLER DAN DI*MONITOR* MENGGUNAKAN KOMPUTER DENGAN SERIAL KOMUNIKASI *ETHERNET*

Nama Mahasiswa 1	**:**	**Ika Hasfritasari**
NRP	**:**	**2213 039 002**
Nama Mahasiswa 2	**:**	**Cahyo Aditya Laksono**
NRP	**:**	**2213 039 031**
Dosen Pembimbing 1	**:**	**Ir. Josaphat Pramudijanto, M.Eng**
NIP	**:**	**19621005 199003 1 003**
Dosen Pembimbing 2	**:**	**Agus Suhanto, S.Pd**
NIP	**:**	**19650821 198603 1 010**

ABSTRAK

Motor induksi tiga fasa merupakan jenis motor yang paling banyak digunakan di bidang industri. Namun motor induksi tiga fasa memiliki kelemahan pada pengontrolan kecepatannya yang hanya bergantung pada tegangan dan frekuensi *input*. Pada Tugas Akhir ini menjelaskan tentang kendali kontrol motor 3 fasa yang ada di industri dengan *inverter* sebagai pengkonversi tegangan masukan menjadi keluaran dalam bentuk frekuensi dan sensor *rotary encoder* sebagai pembaca kecepatan. Metode yang digunakan dalam pengendalian kecepatan adalah dengan menggunakan kontrol *PID* yang ada pada *software LABView*, selain itu motor juga dihubungkan dengan beban untuk mengetahui respon motor terhadap pembebanan, dalam hal ini beban diibaratkan dengan rem *elektromagnetik.* Nilai Kp, Ki, dan Kd *ideal* yang diperoleh dari metode *trial and error* adalah Kp 2,7, nilai Ki 1,7, dan nilai Kd 0,1. Pembebanan mempengaruhi waktu *Rise Time* motor, semakin besar pembebanan maka waktu *Rise Time* akan semakin lama, contohnya pada *setpoint* 1600 *Rise Time* yang awalnya 7,2 *sekon* naik mejadi 8,2 *sekon* ketika diberi beban I dan naik lagi menjadi 8,5 *sekon* ketika diberi beban II.

Kata Kunci : Motor Induksi 3 fasa, *Rotary Encoder*, *PID, LabVIEW*

-----Halaman ini sengaja dikosongkan-----

INDUCTION MOTOR SPEED CONTROL 3 PHASE 1/4 PK WITH MICROCONTROLLER AND MONITORED BY COMPUTER USING ETHERNET SERIAL COMMUNICATION

Name of Student 1	**: Ika Hasfritasari**
Number of Registration	**: 2213 039 002**
Name of Student 2	**: Cahyo Aditya Laksono**
Number of Registration	**: 2213 039 031**
Supervisor 1	**: Ir. Josaphat Pramudijanto, M.Eng**
ID Number	**: 19621005 199003 1 003**
Supervisor 2	**: Agus Suhanto, S.Pd**
ID Number	**: 19650821 198603 1 010**

ABSTRACT

Three phase induction motor is a type of motor most widely used in industry. But the three-phase induction motor has a weakness in controlling the speed that depends only on the input voltage and frequency. This final project describes the control of 3 phase motor control in the industry with an inverter as a voltage converter inputs into outputs in the form of frequency and sensor rotary encoder as a speed reader. The method used in the speed control is to use the existing PID control in LabVIEW software, in addition to the motor is also connected with a load to determine the motor response to the imposition, in this case the load is likened to the electromagnetic brake. Value Kp, Ki, and Kd ideal obtained from the method of trial and error is 2.7 Kp, Ki values of 1.7, 0.1 and Kd values. Imposition of a Rise Time affects the motor, the greater the Rise Time of loading, the time will be longer, for example, the Rise Time setpoint 1600 which was originally a 7.2 second ride form the 8.2 second when given load I and rose again to 8.5 second when given burden II.

Key Words : *3-phase induction motor*, *Rotary Encoder*, *PID, LabVIEW*

-----Halaman ini sengaja dikosongkan-----

DAFTAR ISI

HALAMAN

DAFTAR GAMBAR

HALAMAN

DAFTAR TABEL

HALAMAN

-----Halaman ini sengaja dikosongkan-----

BAB I
PENDAHULUAN

1.1 Latar Belakang

Pada saat ini motor induksi tiga fasa merupakan jenis motor yang paling banyak digunakan di bidang industri dibandingkan motor jenis lain. Ini dikarenakan motor induksi tiga fasa memiliki banyak keunggulan yaitu konstruksinya yang sederhana, tahan lama, perawatan mudah dan efisiensinya tinggi. Tetapi terdapat juga suatu kelemahan dari motor induksi tiga fasa yaitu torsi awal yang rendah serta kesulitan dalam mengatur kecepatan. Sedangkan untuk kebutuhan produksi pengaturan kecepatan motor sangatlah penting karena untuk menyesuaikan dengan proses produksi.

Seringkali dalam aplikasinya motor induksi dituntut untuk bekerja pada kecepatan tertentu. Bertambahnya beban yang diberikan akan memperbesar kopel motor sehingga akan memperbesar arus induksi pada rotor yang menyebabkan *slip* antara medan putar dan putaran rotor pun akan bertambah besar. Perubahan beban ini akan menyebabkan kecepatan putar motor induksi tidak stabil. Untuk meningkatkan efisiensi dan performa motor induksi sistem kontrol sangatlah diperlukan.

Pada tugas akhir ini digunakan motor induksi 3 fasa dengan rem elektromagnetik sebagai *plant* yang akan dikontrol dengan teknik *PID*. Pada motor induksi 3 fasa, perubahan kecepatan dapat diatur dengan cara mengubah-ubah besarnya tegangan inputan yang diberikan pada motor. Serta merancang dan membuat kontroler *PID* dengan meletakkan operasi algoritmanya pada *software LabVIEW*.

Selain itu dibutuhkan sistem monitoring kecepatan motor induksi di industri yang sedang beroperasi agar kinerja motor selalu terpantau dengan baik, sehingga jika terjadi penurunan *performa* motor dapat segera diketahui dan diatasi sehingga tidak sampai menimbulkan kerugian pada perusahaan. Untuk itu pada tugas akhir ini dibuatlah suatu sistem *monitoring* yang memungkinkan operator dapat mengawasi kinerja motor yang sedang beroperasi secara *real time.*

1.2 Rumusan Masalah

Seperti yang kita ketahui, saat ini industri telah berkembang pesat di Indonesia, mesin-mesin produksi yang digunakan pun juga semakin canggih. Motor penggerak mesin produksi di industri telah didominasi dengan penggunaan motor induksi 3 fasa. Namun penggunaan motor induksi memiliki sebuah kelemahan, yakni pembebanan yang berubah akan mempengaruhi kecepatan putar motor. Padahal dalam praktiknya kestabilan kecepatan putar motor sangatlah penting mengingat hal tersebut akan sangat mempengaruhi hasil produksi. Untuk itu dibutuhkan suatu sistem kontrol yang memungkinkan motor berputar dengan kecepatan yang tetap meskipun diberikan pembebanan yang berubah-ubah.

1.3 Batasan Masalah

Dari perumusan masalah di atas, maka batasan masalah dari tugas akhir ini adalah :

1. Motor Induksi yang digunakan hanya terbatas pada spesifikasi motor yang disediakan di lab saja, yaitu motor induksi 3 fasa 1/4 PK.
2. Pengontrolan yang dilakukan hanya terbatas pada pengontrolan kecepatan saja.
3. Sistem kontrol yang digunakan adalah dengan pengaturan *PID* dengan metode *Trial Error.*
4. Sensor yang digunakan sebagai pembaca putaran motor adalah sensor *rotary encoder*.
5. Pembebanan diberikan menggunakan rem elektromagnetik dengan nilai *range* dari *output* keluaran dari *Autotrafo* yaitu 0-240 Volt.
6. Induksi magnetis pada beban tidak diperhitungkan secara matematis.

Dengan adanya batasan masalah ini diharapkan hasil akhir atau tujuan dari Tugas Akhir ini dapat dicapai dengan baik.

1.4 Tujuan

Tujuan kami dalam membuat Tugas Akhir ini adalah sebagai berikut :

1. Merancang dan membuat rangkaian pengontrol kecepatan motor induksi 3 fasa menggunakan mikrokontroler dengan tujuan mempermudah operasi motor induksi di industri karena

kinerja motor induksi dapat dipakai pada berbagai macam kecepatan.

2. Merancang dan membuat rangkaian penstabil kecepatan motor induksi 3 fasa pada industri agar dapat digunakan pada berbagai beban sehingga dapat menyesuaikan dengan proses produksi yang dibutuhkan.

1.5 Sistematika

Sistematika pembahasan tugas akhir ini terdiri dari lima bab, yaitu Pendahuluan, Teori Penunjang, Perancangan dan Pembuatan Alat, Pengujian dan Analisa Alat, serta Penutup.

BAB I : PENDAHULUAN

Membahas tentang latar belakang, permasalahan, batasan masalah, maksud dan tujuan, sistematika laporan, serta relevansi.

BAB II : TEORI PENUNJANG

Membahas tentang teori - teori penunjang yang diperlukan dan dipergunakan sebagai penunjang pengerjaan Tugas Akhir

Bab III : PERANCANGAN DAN PEMBUATAN ALAT

Membahas tentang perencanaan dan pembuatan perangkat keras *(hardware)* yang terdiri dari perancangan elektronik dan perancangan mekanik serta pembuatan dan perancangan perangkat lunak *(software)*.

BAB IV : PENGUKURAN DAN ANALISA ALAT

Membahas tentang pengujian dengan cara pengukuran alat yang terdiri dari pengujian pengukuran perangkat keras dan juga perangkat lunak.

BAB V : PENUTUP

Menjelaskan tentang kesimpulan dari tugas akhir ini dan saran-saran untuk pengembangan alat ini lebih lanjut.

1.6 Relevansi

Relevansi yang kami harapkan dalam membuat Tugas Akhir ini adalah sebagai berikut :

1. Memenuhi kurikulum dan persyaratan kelulusan dari kuliah Diploma 3 (tiga) D3 Teknik Elektro FTI Institut Teknologi Sepuluh Nopember Surabaya
2. Mempermudah operasi motor induksi di industri karena kinerja motor induksi dapat dipakai pada berbagai macam kecepatan dan beban.

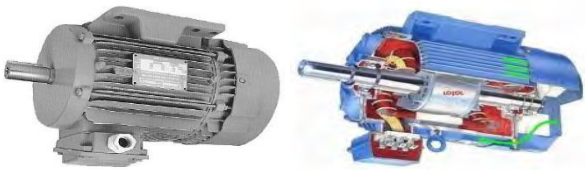
BAB II
TEORI PENUNJANG

Pada bab ini dibahas mengenai teori - teori yang menunjang dalam proses pembuatan alat Tugas Akhir. Teori yang menunjang dalam Tugas Akhir ini diantaranya adalah Motor Induksi 3 Fasa, *Rotary Encoder*, Pengereman Beban Magnetis, *Inverter* 3 Fasa, *Ethernet Board*, serta pengontrolan *PID,* dan Rangkaian MAX232.

2.1 Motor Induksi 3 Fasa [1]

Motor induksi adalah alat penggerak yang paling banyak digunakan dalam dunia industri. Hal ini dikarenakan motor induksi mempunyai kontruksi yang lebih sederhana, kokoh, harganya relatif murah serta perawatannya yang mudah, Sehingga motor induksi mulai menggeser penggunaan motor DC pada industri. Selain keunggulan di atas motor induksi juga memiliki kelemahan yaitu pengaturan motor induksi lebih rumit dari motor DC. Hal ini disebabkan motor induksi memiliki beberapa parameter yang bersifat *non-linier*, tertutama resistansi rotor, yang memiliki nilai bervariasi untuk kondisi operasi yang berbeda.

Pada dasarnya motor induksi dioperasikan pada kecepatan yang konstan, jika beban berubah maka kecepatan motor juga akan berubah. Karena itu untuk mempertahankan agar kecepatan tetap konstan maka tegangan dan frekuensi harus diatur. Namun untuk mengatur tegangan agar didapatkan unjuk kerja yang diharapkan perlu mengatur ulang jumlah kutub stator dan cara lainnya adalah dengan mengubah frekuensi jaringan yang men-*supply* motor tersebut.Hal tersebut diperlukan dengan tujuan antara lain untuk mengurangi besarnya arus start, meredam getaran dan hentakan mekanis saat starting. Karena itu, banyak dilakukan penelitian tentang pengaturan putaran motor induksi tersebut. Salah satunya adalah dengan cara mengubah frekuensi catu daya yang masuk ke motor, untuk mengatur kecepatan motor.

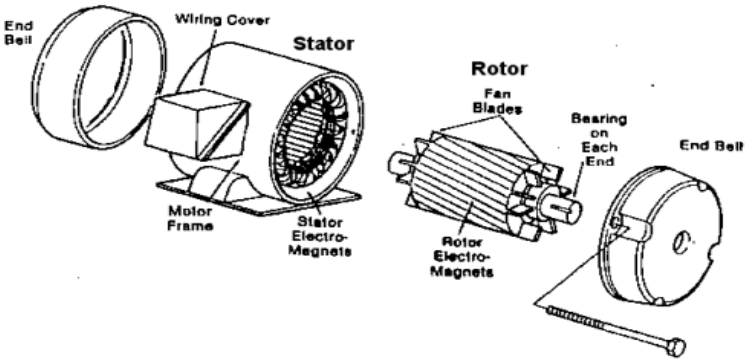


(a) Bentuk Fisik **(b)** Nampak Dalam

Gambar 2.1 Motor Induksi 3-Fasa

2.1.1 Konstruksi Motor AC 3 Fasa

Motor induksi memiliki dua komponen listrik utama yang dapat dilihat pada Gambar 2.2.



Gambar 2.2 Komponen Motor Induksi

1. Rotor adalah bagian dari motor induksi yang bergerak yang berada pada bagian tengah konstruksi motor, berdasarkan konstruksinya rotor dapat diklasifikasikan menjadi dua yaitu:
   a. Rotor sangkar terdiri dari batang penghantar tebal yang dilekatkan dalam petak-petak *slots* paralel. Batang-batang tersebut diberi hubungan pendek pada kedua ujungnya dengan alat cincin hubungan pendek.
   b. Rotor belitan yang memiliki gulungan tiga fase, lapisan ganda dan terdistribusi. Dibuat melingkar sebanyak kutub stator. Tiga fase digulungi kawat pada bagian dalamnya dan ujung yang lainnya dihubungkan ke cincin kecil yang

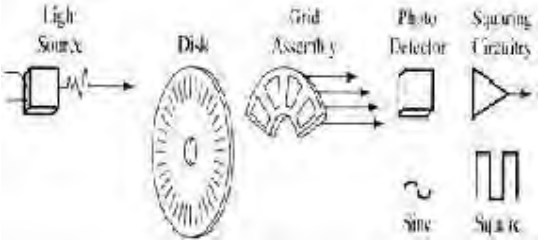
dipasang pada batang as dengan sikat yang menempel padanya.

2. Stator adalah bagian yang diam pada motor induksi dan umumnya berada pada sisi luar motor induksi. Stator dibuat dari sejumlah *stampings* dengan *slots* untuk membawa gulungan tiga fase. Gulungan ini dilingkarkan untuk sejumlah kutub yang tertentu.

2.2 *Rotary Encoder* [2]

Rotary encoder adalah divais elektromekanik yang dapat memonitor gerakan dan posisi. *Rotary encoder* umumnya menggunakan sensor optik untuk menghasilkan serial pulsa yang dapat diartikan menjadi gerakan, posisi, dan arah. Sehingga posisi sudut suatu poros benda berputar dapat diolah menjadi informasi berupa kode digital oleh *rotary encoder* untuk diteruskan oleh rangkaian kendali.

Rotary encoder tersusun dari suatu piringan tipis yang memiliki lubang-lubang pada bagian lingkaran piringan. LED ditempatkan pada salah satu sisi piringan sehingga cahaya akan menuju ke piringan. Di sisi yang lain suatu *photo-transistor* diletakkan sehingga *photo-transistor* ini dapat mendeteksi cahaya dari LED yang berseberangan. Piringan tipis tadi dikopel dengan poros motor, atau divais berputar lainnya yang ingin kita ketahui posisinya, sehingga ketika motor berputar piringan juga akan ikut berputar. Apabila posisi piringan mengakibatkan cahaya dari LED dapat mencapai *photo-transistor* melalui lubang-lubang yang ada, maka *photo-transistor* akan mengalami saturasi dan akan menghasilkan suatu pulsa gelombang persegi. Gambar 2.3 menunjukkan bagan skematik sederhana dari *rotary encoder*. Semakin banyak deretan pulsa yang dihasilkan pada satu putaran menentukan akurasi *rotary encoder* tersebut, akibatnya semakin banyak jumlah lubang yang dapat dibuat pada piringan menentukan akurasi *rotary encoder* tersebut.



Gambar 2.3. Blok Penyusun *Rotary Encoder*

2.2.1 Autonics E50S8-100-3-N-5

Sensor *Rotary Encoder* yang akan digunakan pada tugas akhir ini adalah Sensor *Rotary Encoder* produk dari Autonics dengan tipe E50S8-100-3-N-5 yang berfungsi untuk membaca kecepatan putar motor induksi 3 fasa. Sensor ini memiliki resolusi sebesar 100 ppr (*Pulse per Rotation*) dan memiliki 3 *output phase* yaitu *channel* A, *channel* B, dan *channel* Z. Dimensi dan spesifikasi elektris dari *RotaryEncoder* lebih jelasnya dapat dilihat pada Gambar 2.4. dan Tabel 2.1 dibawah ini.

Gambar 2.4. Sensor *Rotary Encoder* Autonics E50S8-100-3-N-5

Tabel 2.1 Spesifikasi *Rotary Encoder* Autonics E50S8-100-3-N-5 [7]

Type Encoder	*Incremental*
Tegangan *Supply*	5 VDC (*Ripple p-p*); maksimal 5%
Konsumsi Arus	20 mA DC (maksimal)
Resolusi	100 ppr (*Pulse per Rotation*)
Output Phase	A,B,Z
Kontrol *Output (Output Type)*	NPN *Open Collector Output*
Kontrol *Output (Load Current)*	50 mA max.

2.3 Beban Pengereman Magnetis [3]

Rem adalah suatu alat yang digunakan untuk melakukan suatu aksi yang akan menurunkan kecepatan dalam selang waktu yang ditentukan. Tipe rem yang umum digunakan adalah rem yang menggunakan gaya gesek untuk memberikan gaya lawan terhadap gaya gerak. Sistem pengereman elektromagnetik menggunakan gaya elektromagnetik untuk memperlambat suatu gerakan. Sebuah piringan dengan bahan logam non-feromagnetik terpasang dengan poros yang berputar. Piringan tersebut

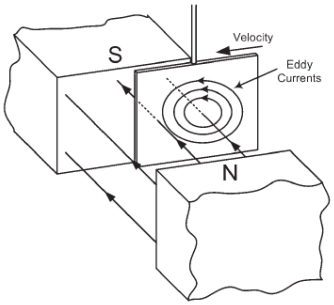
diapit oleh sisi stator berupa sistem lilitan elektromagnetik yang dapat membangkitkan medan magnet dari aliran listrik.

Arus listrik menimbulkan medan magnet pada lilitan dan logam piringan yang memotong medan magnet tersebut akan menimbulkan arus *eddy* pada piringan itu sendiri. Arus *eddy* ini akan menimbulkan medan magnet yang arahnya berlawanan dengan medan magnet sebelumnya, sehingga menghambat gerakan putar dari poros tersebut. Rem elektromagnetik akan optimal untuk memberikan penurunan kecepatan, bukan untuk menghentikan gerak suatu objek. Sehingga Rem ini sering diaplikasikan untuk sistem pengereman pada *roller coaster*, kereta api dan juga digunakan pada alat *dinamometer* untuk pengukuran torsi suatu mesin.Contoh dari bentuk fisik dari rem elektromagnetik ditunjukkan oleh Gambar 2.5. Arus *eddy* yang melingkar menyebabkan medan magnet induksi melawan arah medan magnet mula-mula. Hal ini menyebabkan gaya pengereman yang melawan arah kecepatan konduktor yang bergerak memotong medan magnet dari kedua solenoid.

Gambar 2.5. Bentuk Fisik Rem Elektromagnetik

Gaya pengereman yang dihasilkan oleh arus melingkar *eddy* ditunjukkan oleh Gambar 2.6. Medan magnet yang arahnya menjauhi pengamat. Kemudian sebuah konduktor memotong medan magnet tersebut dengan kecepatan (besar dan arah) tertentu. Berdasarkan hukum faraday, apabila terjadi perubahan medan magnet, maka akan timbul ggl pada konduktor. Pada konduktor, bidang yang mengalami perubahan fluks magnet hanya pada kedua sisinya, yang pertama adalah saat keluar dari medan maget (fluks magnet yang lewat pada konduktor berkurang) dan yang kedua adalah saat memasuki medan magnet (fluks magnet yang melewati konduktor bertambah). Sedangkan bagian tengah konduktor

tidak mengalami perubahan fluks magnet sehingga tidak timbul lagi. Dengan artian, gaya lawan hanya dihasilkan apabila permukaan tersebut memiliki kecepatan. Semakin tinggi kecepatan maka gaya lawan yang dihasilkan juga semakin besar. Namun semakin rendah kecepatan, maka gaya lawan akan semakin kecil.



Gambar 2.6 Gaya Pengereman Arus *Eddy*

Medan magnet yang arahnya menjauhi pengamat. Kemudian sebuah konduktor memotong medan magnet tersebut dengan kecepatan (besar dan arah) tertentu. Berdasarkan hukum faraday, apabila terjadi perubahan medan magnet, maka akan timbul ggl pada konduktor. Pada konduktor, bidang yang mengalami perubahan fluks magnet hanya pada kedua sisinya, yang pertama adalah saat keluar dari medan maget (fluks magnet yang lewat pada konduktor berkurang) dan yang kedua adalah saat memasuki medan magnet (fluks magnet yang melewati konduktor bertambah). Sedangkan bagian tengah konduktor tidak mengalami perubahan fluks magnet sehingga tidak timbul lagi. Dengan artian, gaya lawan hanya dihasilkan apabila permukaan tersebut memiliki kecepatan. Semakin tinggi kecepatan maka gaya lawan yang dihasilkan juga semakin besar. Namun semakin rendah kecepatan, maka gaya lawan akan semakin kecil.

2.4 *Inverter Siemens Sinamics G110* [3]

Inverter / variable frequency drive / variable speed drive merupakan sebuah alat pengatur kecepatan motor dengan mengubah nilai frekuensi dan tegangan yang masuk ke motor. pengaturan nilai frekuensi dan tegangan ini dimaksudkan untuk mendapatkan kecepatan putaran dan torsi motor yang di inginkan atau sesuai dengan kebutuhan. Secara

sederhana prinsip dasar *inverter* untuk dapat mengubah frekuensi menjadi lebih kecil atau lebih besar yaitu dengan mengubah tegangan AC menjadi tegangan DC kemudian dijadikan tegangan AC lagi dengan frekuensi yang berbeda atau dapat diatur.

Inverter siemens sinamics G110 adalah *inverter* pengendali frekuensi untuk putar balik motor AC tiga fasa *Inverter* ini adalah sebuah *mikroprocessor* pengontrol yang menggunakan teknologi *Insulated Gate Bipolar Transistor (IGBT)* yang membuatnya lebih memiliki kegunaan dan keandalan. Bentuk fisik *inverter siemens sinamics g110* dapat dilihat pada Gambar 2.7.

Gambar 2.7 Bentuk Fisik *Inverter Siemens Sinamics G110*

Spesifikasi *Inverter* SINAMICS G110 [5]
Input Voltage : 220 – 240 Volt
Power Range : 0,12 kW – 3,0 kW
Input Frequency : 47 Hz – 63 Hz
Output frequency : 0 Hz – 650 Hz
Cos Phi : ≥ 0,95

Sinamics G110 hanya memiliki satu pilihan *mode operator panel*, yaitu:

Basic Operator Panel (BOP)

Pada *mode BOP* yang ditampilkan hanya berupa informasi frekuensi (Hz),

Parameter dan informasi yang ditampilkan *LCD*. Pada Tugas Akhir ini, *inverter Sinamics G110* digunakan dalam *mode BOP*.Dapat dilihat pada Gambar 2.8.

Gambar 2.8. Operator *Panel BOP*

Terdapat beberapa tombol yang memiliki fungsi berbeda untuk pengoperasian *inverter sinamics G110*. Beberapa tombol tersebut dapat dilihat pada Tabel 2.2

Tabel 2.2 Fungsi Tombol *Inverter Micromaster G110* [5]

Panel/Button	**Fungsi**	**Keterangan**
	Status Indikasi	Tampilan *LCD* saat akan memilih parameter yang akan di-*setting*
	*Start* Motor	Tombol untuk menjalankan motor
	*Stop Motor*	Tombol untuk memberhentikan motor
	*Change Direction*	Tombol untuk mengubah arah putaran motor
	*Jog Motor*	Tombol untuk menjalankan motor sesuai *joging present*. Motor akan berjalan selama tombol ditahan

Panel/Button	Fungsi	Keterangan
Fn	*Functions*	Tombol untuk menampilkan informasi tambahan. Tombol ini
P	*Access Parameter*	Tombol untuk mengakses parameter
▲	*Increase Value*	Tombol untuk menaikkan nilai yang ditampilkan
▼	*Decrease Value*	Tombol untuk menurunkan nilai yang ditampilkan

Dan untuk mengakses parameter pada *inverter* ini digunakan tombol-tombol sebagai berikut :

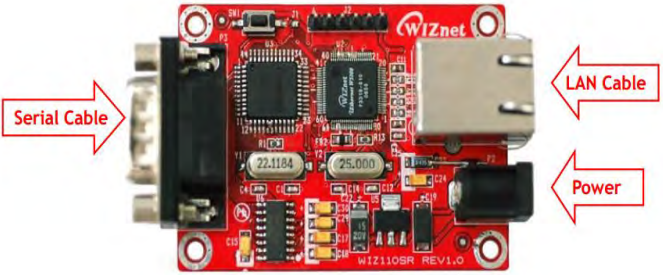
Tabel 2.3 Fungsi Tombol *Inverter* [5]

	Langkah-langkah	Tampilan
1	Tekan P untuk mengakses parameter	r0000
2	Tekan ▲ hingga muncul P0003	P0003
3	Tekan P untuk mengubah nilai parameter	1
4	Tekan ▲ atau ▼ untuk memilih nilai yang diinginkan	3
5	Tekan P kembali untuk memilih	P0003
6	Sekarang pengguna bias mengatur parameter	

2.5 Ethernet Board WIZ110SR [6]

WIZ110SR adalah gateway modul yang mengkonversi protokol RS-232 ke TCP / IP protokol. Ini memungkinkan jauh mengukur, mengelola dan mengendalikan perangkat melalui jaringan berbasis pada Ethernet dan TCP / IP dengan menghubungkan ke peralatan yang ada dengan serial RS-232 interface. Dengan kata lain, WIZ110SR merupakan

sebuah protokol konverter yang mentransmisikan data yang dikirim dengan serial peralatan sebagai TCP / IP tipe data dan mengkonversi kembali TCP / IP data yang diterima melalui jaringan ke data serial untuk mengirimkan kembali ke peralatan. Untuk tampilan *hardware Ethernet Board WIZ110SR* dapat dilihat pada Gambar 2.9



Gambar 2.9. *Ethernet Board* WIZ110SR

Untuk spesifikasi *Ethernet Board* WIZ110SR lebih lengkapnya dapat dilihat pada tabel berikut ini :

Tabel 2.4 Spesifikasi *Ethernet Board* WIZ110SR [6]

***Input* Tegangan**	DC 5 V
Konsumsi Arus	Dibawah 180 mA
TCP/IP	W5100 (Ethernet MAC & PHY Tertanam)
Protokol	TCP, UDP, IP, ARP, ICMP, MAC, DHCP, PPPoE, DNS
Network Interface	10/100 Mbps (Auto detection), RJ-45 Connector
Serial Interface	RS232 (DB9)
Serial Sinyal	TXD, RXD, RTS, CTS, GND
Data Bits	7,8
Speed	Up to 230 Kbps

2.6 Sistem Kontrol PID [4]

PID dari singkatan *(Propotional-Integral-Derivative* kontrol*ler*) merupakan kontroler untuk menentukan presisi suatu sistem instrumentasi dengan karakteristik adanya umpan balik pada sistem tersebut. Komponen Kontrol PID ini terdiri dari tiga jenis yaitu *Propotional, integral,* dan *derivative*.

Kontrol PID merupakan kombinasi dari tiga jenis kontroler. Jika masing-masing dari ketiga jenis kontroler tersebut berdiri sendiri maka hasil yang dicapai akan kurang baik, sebab masing-masing memiliki kelebihan dan kelemahan sendiri. Kombinasi dari ketiga jenis kontroler tersebut menjadi satu diharapkan mampu memberikan kontribusi dari kelebihan masing-masing. PID dari singkatan *(Propotional-Integral-Derivative* kontrol*ler*) merupakan kontroler untuk menentukan presisi suatu sistem instrumentasi dengan karakteristik adanya umpan balik pada sistem tersebut. Komponen kontrol PID ini terdiri dari tiga jenis yaitu *Propotional, integral,* dan *derivative*. Ketiganya dapat dipakai bersamaan maupun sendiri-sendiri tergantung dari respon yang kita inginkan terhadap suatu plant. Kontroler PID dibagi menjadi dua, yakni *PID Continous* dan *PID Discrete.* Berikut akan dibahas tentang dua buah kontroler PID tersebut.

2.6.1 Kontroler PID *Continous*

Pada bagian ini akan dibahas tentang kontroler P, kontroler I, dan kontroler D. pada kontroler *PID Continous* ini respon motor selalu memiliki nilai pada setiap waktu, karena sinyal yang terbentuk selalu membuat sinyal kontinyu. Kontroler P digunakan sebagai perintah pada motor untuk cepat mencapai *Rise Time*. Pada kontroler P ini bisa mempengaruhi sistem yaitu menambah atau mengurangi kestabilan, dapat memperbaiki respon *transien* khususnya adalah *Rise Time* dan *Time Settling.* Kontroler I digunakan untuk mengurangi *eror steady*, respon motor lebih lambat karena dapat mencapai *stady state*, dan juga pada kontroler I dapat menambah ketidakstabilan karena menambah orde pada sistem. Sedangkan kontroler D hanya digunakan apabila terjadi sinyal eror Karena kontroler D ini bekerja apabila ada eror, memberikan efek redaman pada sistem yang berisolasi sehingga bisa memperbesar pemberian nilai Kp, serta juga dapat untuk memperbaiki respon transien karena bekerja pada saat eror. Oleh karena itu, kontroler D ini tidak dapat berdiri sendiri dalam penggunaanya.

Apabila menggunakan konroler P saja, maka respon motor akan cepat untuk mencapai *Rise Time* akan tetapi akan lama untuk mencapai *stady state* karena kontroler P ini member pengaruh langsung (sebanding) pada eror. Semakin besar eror, semakin besar sinyal kendali yang dihasilkan kontroler. Berikut adalah perumusan dari kontroler P:

$$P = Kp = \frac{\tau}{K.\tau} \quad \text{.....(2.1)}$$

Apabila menggunakan PI, maka respon motor akan cepat untuk mencapai *Rise Time* dan juga akan mencapai *stady state* karena kontroler I ini bersifat untuk mengurangi eror *stady state* walaupun dengan waktu yang lama untuk mencapai *stady state*nya. Berikut adalah perumusan kontroler PI :

$$PI = Kp\ (1+\frac{1}{\tau_i . s}) \quad \text{.....(2.2)}$$

Apabila menggunakan PD, maka respon motor akan cepat mencapai *rise time* dan langsung bisa mencapai *stady state* akan tetapi respon tidak stabil karena hanya dapat dipergunakan kalau terdapat eror saja. Berikut adalah perumusan kontroler PD (orde 2):

$$P(t) = Kp\ (e(k) + \tau_d \frac{de(\tau)}{dt}). \quad \text{.....(2.3)}$$

Apabila menggunakan kontroler PID maka respon motor akan cepat untuk mencapai *rise time* dan juga cepat mencapai *steady state* karena kontroler mengurangi eror *steady state*, sedangkan kontroler D ini bisa bekerja karena terdapat eror dari kontroler P dan kontroler I sehingga motor dapat mencapai kondisi yang stabil. Perumusan dan kontroler PID *Continous* ini adalah sebagai berikut:

$$U(s) = Kp\ \{1 + \frac{1}{\tau_i s} + \tau_d s\} \quad \text{.....(2.4)}$$

2.6.2 Kontroler PID *Diskrit*

Pada bagian ini akan membahas tentang kontroler P, kontroler I, dan kontroler D. Pada kontroler PID *Discrete* ini respon motor memili nilai pada waktu-waktu tertentu karena sinyal yang terbentuk selalu membentuk sinyal *diskrit*. Kontroler P memberikan kesalahan

stasioner dalam semua kasus kecuali ketika sistem kontrol *input* adalah nol dan sistem proses nilai sama dengan nilai yang diinginkan. Kontroler I memberikan tambahan dari jumlah kesalahan sebelumnya ke sistem kontrol *input*. Menjumlahkan kesalahan akan terus sampai nilai proses sistem sama dengan volume yang diinginkan dan ini mengakibatkan tidak ada kesalahan stasioner ketika referensi stabil. Sedangkan kontroler D memberikan tambahan dari laju perubahan dalam kesalahan sistem kontrol *input* atau bisa juga dibilang kalau kontroler D ini akan bekerja apabila ada eror. Perubahan cepat pada kesalahan akan memberikan tambahan ke sistem kontrol *input*. Oleh karena itu, kontroler D ini tidak dapat berdiri sendiri dalam penggunaanya. PID *Discrete* ini untuk lebih jelasnya dapat dilihat pada lampiran B.

Apabila menggunakan kontroler P saja, maka respon motor tidak akan mencapai *rise time* sehingga sistem tidak akan stabil. Berikut adalah perumusan kontroler P :

$$P = Kp\ e(n).\Delta T \quad \text{.....(2.5)}$$

Apabila menggunakan kontroler PI, maka respon motor akan melambai dan akan sering terjadi isolasi. Sehingga motor akan lambat untuk mencapai keadaan *steady state*. Berikut adalah perumusan kontroler PI :

$$PI = Kp\ e(n) + Ki \sum_{k=0}^{n} e(k).\Delta T \quad \text{.....(2.6)}$$

Apabila menggunakan PD, maka motor akan memberikan nilai proses sistem naik lebih cepat daripada kontroler P. berikut ini adalah perumusahan kontroler PD :

$$PD = Kp\ e(n) + Kd\big(e(n) - e(n-1)\big).\Delta T \quad \text{.....(2.7)}$$

Apabila menggunakan kontroler PID maka motor akan memberikan kinerja terbaik. Karena kontroler PI meningkatkan kontroler P dengan menghapus kesalahan stasioner dam kontroler PID meningkatkan kontroler PI dengan respons yang lebih cepat dan tidak *overshoot*. Perumusan dari kontroler PID *discrete* ini adalah sebagai berikut :

$$U(n) = Kp\ e(n) + Ki \sum_{k=0}^{n} e(k) + Kd\big(e(n) - e(n-1)\big).\Delta T \quad \text{.....(2.8)}$$

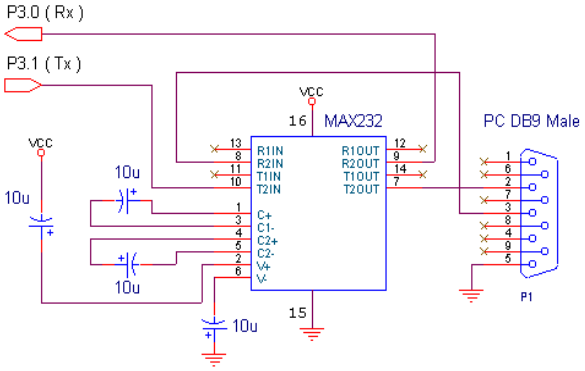
2.7 Rangkaian MAX232 [8]

Pada Mikrokontroler ATMega16 terdapat pin untuk komunikasi Tx dan Rx.Tx digunakan untuk mengirimkan data secara serial sedangkan Rx digunakan untuk menerima data serial. Komunikasi data serial pada mikrokontroler ini masih menggunakan sinyal TTL atau *Transistor Transistor Logic* yaitu sinyal yang gelombang datanya antara 0 dan 5 V. Dengan fasilitas Rx dan Tx ini mikrokontroler bisa komunikasi secara serial baik antar *device* atau dengan komputer.

Jika ingin digunakan untuk berkomunikasi antar *device* tinggal langsung saja hubungkan *cross* antar dua *device* tadi maksudnya Tx *device* 1 dihubungkan ke Rx *device* 2 dan Rx *device* 1 dihubungkan ke Tx *device* 2.

Jika ingin digunakan untuk berkomunikasi dengan komputer maka Rx dan Tx tidak bisa langsung dihubungkan begitu saja karena sinyal yang digunakan berbeda. Komunikasi serial komputer menggunakan sinyal RS232 yaitu sinyal yang gelombangnya antara +25 V sampai -25 V. Oleh karena itu jika ingin diharapkan terjadi komunikasi antara Mikrokontroler dengan komputer dibutuhkan sebuah komponen yang dapat mengubah sinyal level TTL dari mikrokontroler menjadi sinyal level RS232. Salah satu komponen yang sering digunakan adalah IC MAX232.

Supaya dapat digunakan dengan baik maka IC Max232 ini membutuhkan beberapa komponen tambahan yaitu Kapasitor elektrolit (ELCO) sebanyak 4 buah sebesar 1uF. Untuk lebih lengkapnya rangkaian MAX232 dapat dilihat pada Gambar 2.10.



Gambar 2.10 Rangkaian MAX232

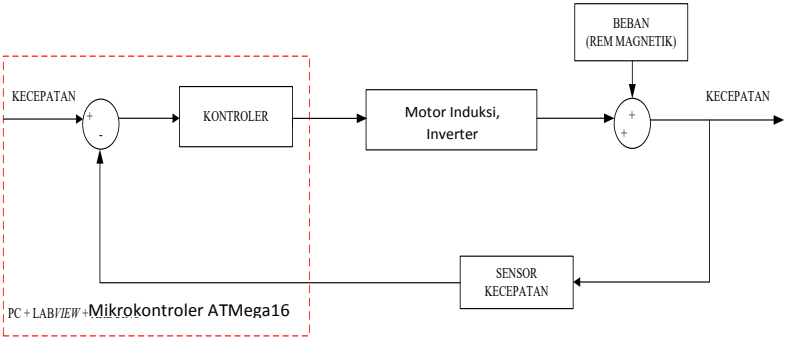
BAB III
PERANCANGAN DAN PEMBUATAN ALAT

Pada tugas akhir yang membuat motor 3 *phasa* berbeban elektromagnetik, ada 2 tahapan yang dilakukan yaitu :

1. Perancangan Perangkat Keras *(Hardware)*
   a. Perancangan *Power Supply*
   b. Perancangan Pengaturan Kecepatan Motor Induksi
   c. *Wiring Diagram Inverter* Sinamics G110
   d. Perancangan *Digital to Analog Converter*
   e. Perancangan Pembacaan Kecepatan oleh Sensor *Rotary Encoder*
   f. Perancangan *Panel Box*
2. Perancangan Perangkat Lunak *(Software)*
   a. Inisialisasi *Inverter* Sinamics G110
   b. *Setting IP* Rangkaian *Ethernet* WIZ110SR
   c. Inisialisasi *Port* LCD 2x16
   d. Perancangan *Software* Pembacaan Kecepatan
   e. Perancangan *Software* Pengaturan Kecepatan
   f. Tampilan *Software LabVIEW*

3.1 Blok Fungsional Sistem

Perancangan sistem dalam pembuatan alat ini secara garis besar disertai urutan dan cara kerja alat ini di ilustrasikan pada Gambar 3.1.



Gambar 3.1 Blok *Diagram* Sistem *Close Loop* PID

Dari Gambar 3.1 dapat dilihat bahwa sistem terdiri dari beberapa blok fungsional yaitu;

1. ATMega16, merupakan mikrokontroler yang berfungsi sebagai *interface* dari perangkat elektronik dan dapat menyimpan program didalamnya.
2. *Inverter,* sebagai penggerak motor induksi AC 3 fasa.
3. Motor Induksi 3 Fasa, digunakan sebagai objek yang akan dikontrol kecepatannya.
4. Rem magnetik, digunakan sebagai media pengereman dan juga sebagai beban bagi motor induksi 3 fasa.
5. Sensor kecepatan (*rotary encoder*), digunakan untuk mengetahui sampai seberapa kecepatan motor yang terjadi.
6. Lab*View*, merupakan perangkat lunak yang digunakan sebagai media untuk memberi masukan atau interupsi ke kontroler.
7. Tampilan (*Display*), untuk tampilan digunakan PC*,* PC ini digunakan untuk menampilkan kecepatan motor induksi 3 fasa.

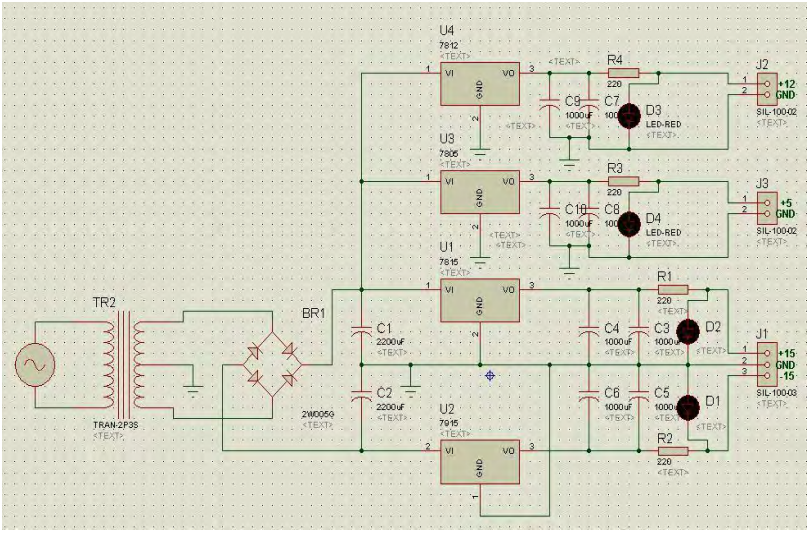
3.2 Perancangan Perangkat Keras

Perancangan Perangkat keras ini dimaksudkan agar dalam pelaksanaan pengerjaan tugas akhir menjadi lebih mudah dan cepat karena setiap detail perangkat keras (*Hardware)* yang akan dikerjakan baik dari sisi elektronik maupun mekanik telah direncanakan secara terperinci dan baik.

3.2.1 Perancangan *Power Supply*

Rangkaian Power *Supply* ini digunakan untuk merubah tegangan VAC dari PLN sebesar 220 V menjadi tegangan searah (VDC) yang aman untuk rangkaian mikrokontroler serta rangkaian elektronik lainnya yang juga membutuhkan Tegangan *Input*an VDC.

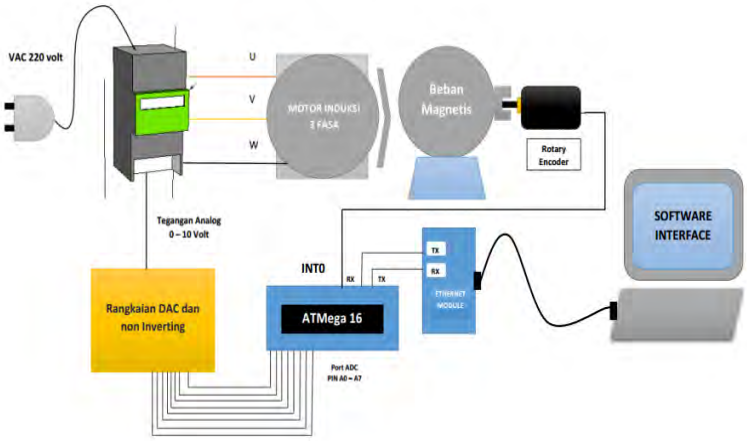
Tegangan Jala-Jala PLN sebesar 220 V diturunkan oleh Trafo atau *Transformator* penurun tegangan menjadi tegangan sesuai yang dibutuhkan yakni 5 Volt (untuk mikrokontroler dan *rotary encoder)* dan 15 Volt (*supply IC DAC)* menggunakan konsep perbandingan lilitan. Karena Tegangan *output*an dari Trafo masih berbentuk Tegangan VAC maka harus disearahkan oleh rangkaian penyearah berupa 4 buah dioda yang berfungsi untuk meloloskan yang mulanya tegangan bolak-balik (2 arah) menjadi tegangan satu arah saja, lalu kapasitor disini berfungsi untuk menghilangkan riak sehingga tegangan yang dihasilkan murni. Rangkaian perancangan *Power Supply* dapat dilihat pada Gambar 3.2.



Gambar 3.2 Rangkaian *Proteus Power Supply*

3.2.2 Perancangan Pengaturan Kecepatan Motor Induksi

Sebagaimana diketahui bahwa pengaturan kecepatan putaran motor induksi dapat dilakukan dengan dua cara yaitu dengan mengatur frekuensi, dan mengubah tegangan masukan dengan mengatur jumlah kutub. Dalam perancangan alat pengatur kecepatan yang di bahas dalam Tugas akhir yakni dengan cara mengatur Frekuensi masukan yang masuk ke motor. Yaitu dengan cara mengatur tegangan masukan ke *inverter* yang kemudian tegangan ini akan dikonversikan oleh *inverter* menjadi outputan frekuensi dengan *range* 0 sampai 50 Hz. Perancangan alat pengatur kecepatan motor induksi tiga fasa dengan mengatur tegangan masukan ke *inverter* ini membutuhkan beberapa peralatan yang akan digunakan yaitu, *Inverter Sinamics* G110 , Mikrokontroler ATMega 16 serta rangkaian *Digital to Analog Converter* dan Rangkaian *Non inverting Amplifier*, dan *Ethernet* sebagai media komunikasi antara sistem dengan *interface* berupa komputer operator.



Gambar 3.3 Perancangan Pengaturan Kecepatan Motor Induksi

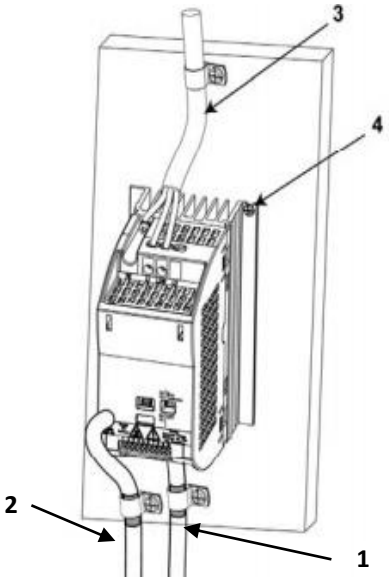
Pengaturan level kecepatan pada Gambar 3.3 dilakukan pada komputer operator, lalu perintah tersebut akan dikirimkan ke rangkaian kontrol mikrokontroler melalui media komunikasi *Ethernet*, dari titik *Transmitter* (Tx) menuju titik *Receiver* (Rx) pada mikrokontroler, data kiriman berupa bilangan *biner* dari komputer ini kemudian diolah dan diterjemahkan oleh mikrokontroler menjadi pulsa-pulsa *digital* 8 bit (Pin A0 sampai A7) yang akan dikonversikan oleh rangkaian DAC dan penguat *non inverting* menjadi suatu keluaran berupa tegangan *analog* dengan *range* antara 0 sampai 10 Volt, tegangan *analog* ini kemudian akan menjadi *input*an pada *inverter* sinamics G110 untuk kemudian akan dijadikan perintah untuk memutar motor sesuai dengan kecepatan yang dikehendaki dengan cara mengubah frekuensi masukan ke motor. Tegangan 0 sampai 10 Volt dari DAC akan dikonversi menjadi outputan frekuensi dengan range 0 sampai 50 Hz.

3.2.3 *Wiring Diagram Inverter* Sinamics G110

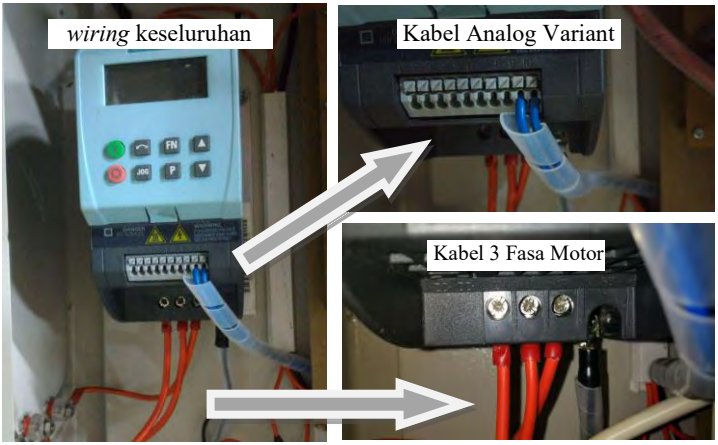
Untuk mengaktifkan *Inverter Sinamics G110* ada beberapa kabel yang harus dikoneksikan. Karena *Inverter* ini diproduksi khusus untuk driver motor-motor induksi industri yang tentunya berdaya besar jadi pengkoneksian atau *wiring diagram* perlu diperhatikan demi keamanan dalam pengoperasian maka dari itu penempatan, dan kerapian kabel

juga harus diperhitungkan dengan baik. *Wiring diagram* pada *Inverter Sinamics G110* lebih jelasnya dapat dilihat pada Gambar 3.4 dan 3.5. Keterangan pengkoneksian kabel pada Gambar 3.4 :

1. Kabel 3 Fasa (U,V,W) ke Motor Induksi
2. Kabel Kontrol *Inverter Sinamics G110* (dapat dilihat pada Gambar 3.6)
3. Kabel Catu Daya *Inverter*
4. Baut penyatu *Inverter* dengan *Box Panel* untuk *grounding*

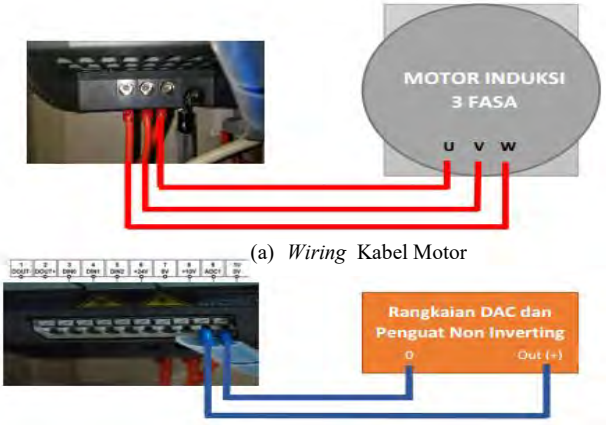


Gambar 3.4 *Wiring Diagram Inverter Sinamics G110* [5]



Gambar 3.5 *Hardware Wiring Diagram Inverter Sinamics G110*

Karena menggunakan pengontrolan dengan mikrokontroler yang telah dihubungkan dengan rangkaian DAC yang *output*nya berupa tegangan *analog* dengan *range* antara 0-10 V maka pin *inverter* yang harus disambung ke mikrokontroler adalah pin ADC yaitu pin 9 dan 10. Untuk *wiring diagram* kabel inverter ke motor dan konfigurasi pin *analog variant* dapat dilihat pada Gambar 3.6.



(a) *Wiring* Kabel Motor

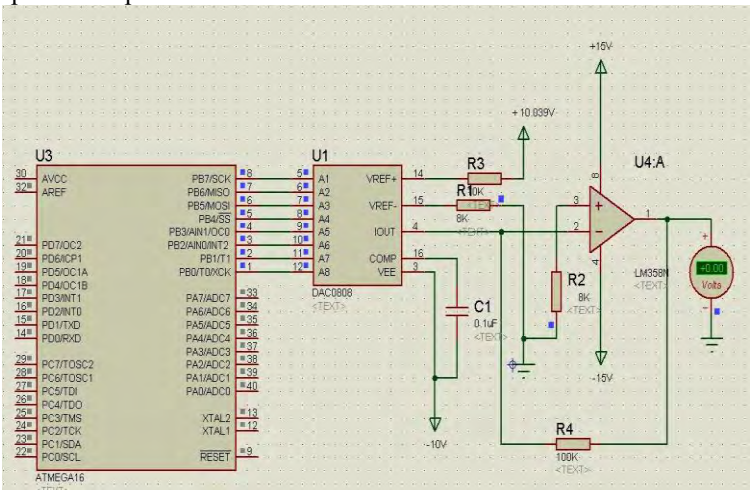
(b) *wiring Analog Variant* ke *DAC*

Gambar 3.6 Konfigurasi Pin Kontrol *Inverter Sinamics* G110

3.2.4 Perancangan *Digital to Analog Converter*

Rangkaian *Digital to Analog Converter* ini dibuat untuk mengkonversikan tegangan *output*an berupa bilangan *digital* biner 8 bit dari Mikrokontroler ATMega 16 menjadi nilai tegangan *analog* yang kemudian menjadi nilai *input*an pada *Inverter.* Rangkaian *DAC* ini juga dilengkapi dengan rangkaian penguat *non inverting. Range* tegangan yang masuk ke *inverter* antara tegangan 0 sampai 10 Volt. Kemudian tegangan *analog* yang masuk ke *inverter* ini akan diubah oleh *inverter* menjadi *output*an frekuensi antara 0 sampai 50 Hz.

DAC (*Digital to Analog Convertion*) adalah perangkat atau rangkaian elektronika yang berfungsi untuk mengubah suatu isyarat *digital* (kode-kode biner) menjadi isyarat *analog* (tegangan *analog*) sesuai harga dari isyarat *digital* tersebut. DAC *(Digital to Analog Convertion)* dapat dibangun menggunakan penguat penjumlah inverting dari sebuah *operasional amplifier* (*Op-Amp*) yang diberikan sinyal *input* berupa data logika *digital* (0 dan 1). Rangkaian *Op-Amp* berfungsi untuk menguatkan tegangan *output*an dari mikrokontroler sebesar 0-5 Volt menjadi tegangan yang dibutuhkan oleh *inverter* yakni sebesar 0-10 Volt.Gambar perancangan Rangkaian DAC dan Penguat *Non Inverting* dapat dilihat pada Gambar 3.7.



Gambar 3.7 Rangkaian *Proteus Digital to Analog Converter*

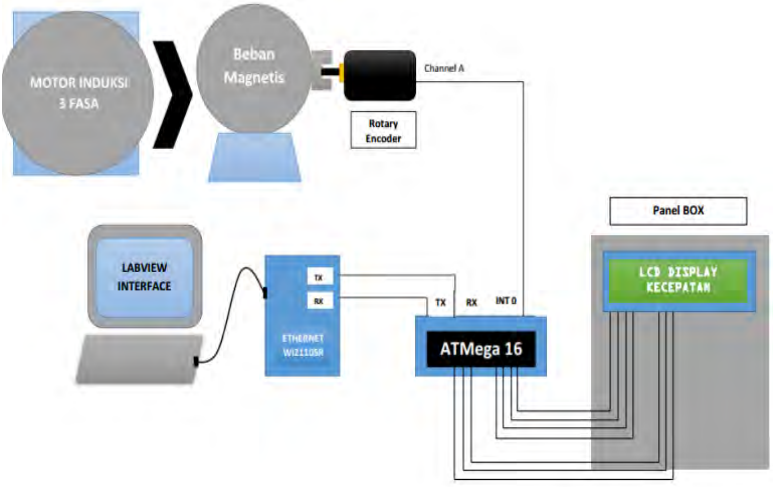
3.2.5 Perancangan Pembacaan Kecepatan oleh *Rotary Encoder*

Untuk mengukur kecepatan putar motor secara *real time* kita akan menggunakan sensor kecepatan *rotary encoder* keluaran Autonics dengan tipe E50S8-100-3-N-5. Hasil pembacaan kecepatan ini selanjutnya akan ditampilkan pada tampilan *interface LabVIEW* pada Komputer dan Tampilan LCD 2x16 yang dipasang pada *Panel Box.*

3.2.5.1 Perancangan Pembacaan Kecepatan

Untuk Pembacaan kecepatan motor dikopel dengan beban elektomagnetis sehinggga ketika motor berputar maka beban magnetis juga akan ikut berputar. Sensor *rotary encoder* yang dipasang pada beban magnetis akan menghitung berapa kecepatan putar motor dengan mengirimkan sinyal berupa pulsa per putaran kepada mikrokontroler, data sinyal pulsa ini kemudian akan diolah oleh mikrokontroler menjadi informasi berupa kecepatan putar motor yang kemudian akan dikirimkan untuk ditampilkan pada *LabVIEW Interface* pada komputer dan LCD *display* kecepatan pada *panel box* melalui pin *Transmitter* Mikrokontroler dan pin *Receiver* pada Modul *Ethernet.*

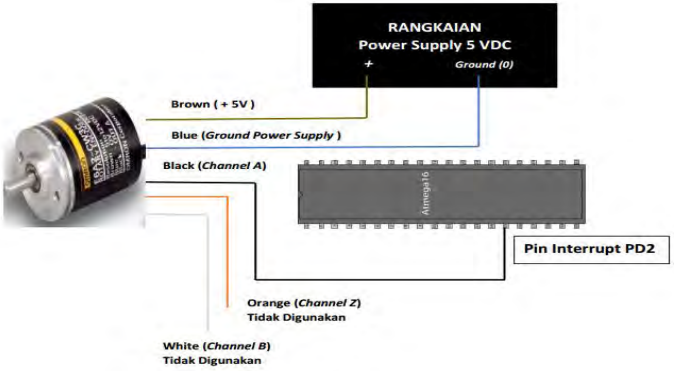
Untuk lebih jelasnya bisa dilihat pada *diagram* blok pada Gambar 3.8.



Gambar 3.8 Perancangan Pembacaan Kecepatan Motor Induksi

3.2.5.2 *Wiring* Sensor *Rotary Encoder* E50S8-100-3-N-5

Terdapat 5 kabel koneksi sensor *rotary encoder* E50S8-100-3-N-5 yang memiliki warna dan fungsi berbeda-beda, yaitu *channel* A (*black*), *channel* B (*white*), *channel Z (orange), supply* +5V (*brown*), dan *ground* (*blue*). Kabel biru dan coklat dihubungkan ke rangkaian *supply* daya sebesar 5 VDC, untuk kabel hitam *(Output Channel A)* ini dihubungkan ke modul *High Speed Counter* atau PIN *Interrupt* yang ada pada Mikrokontroler ATMega16 yaitu pin D2, sedangkan untuk kabel putih *channel* B dan *Channel Z* tidak perlu digunakan karena sensor *rotary encoder* hanya akan digunakan untuk menghitung kecepatan putar motor saja. Untuk lebih jelasnya *wiring diagram* antara kabel *rotary encoder* dan ATMega16 dapat dilihat pada Gambar 3.9 dan Tabel 3.1.



Gambar 3.9 *Wiring* Kabel Sensor *Rotary Encoder*

Tabel 3.1 Koneksi Kabel Sensor *Rotary Encoder* E50S8-100-3-N-5

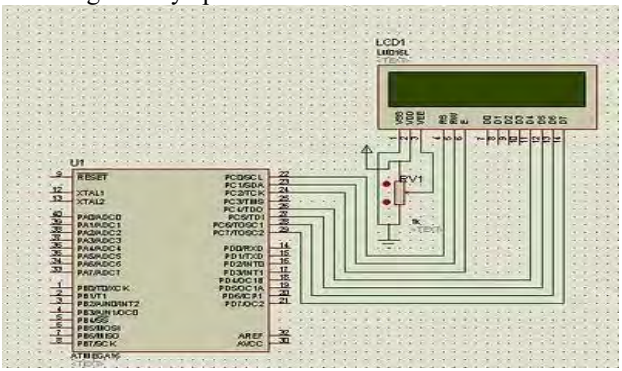
Kabel Sensor	Koneksi
Black (Channel A)	Pin *Interrupt* ATMega16 **(PD2)**
Brown (*Input* 5 V)	*Input Power Supply*
Blue (*Ground*)	*Ground Power Supply*

3.2.5.3 *Wiring* LCD 2x16

Display yang kita akan gunakan pada *project* Tugas Akhir ini adalah *display* LCD 2×16 , artinya LCD terdiri dari 2 baris dan 16 karakter. Bentuk fisiknya dapat dilihat pada Gambar 3.10.

Gambar 3.10 Konfigurasi PIN LCD 2x16

Sebelum membuat programnya terlebih dahulu kita membuat rancangan *hardware* nantinya disini akan digunakan *PORT*C pada ATMega16. Rangkaiannya pada Gambar 3.11.



Gambar 3.11 Rangkaian *Proteus Wiring* antar *Pin* LCD 16x2 dengan Mikrokontroler ATMega16

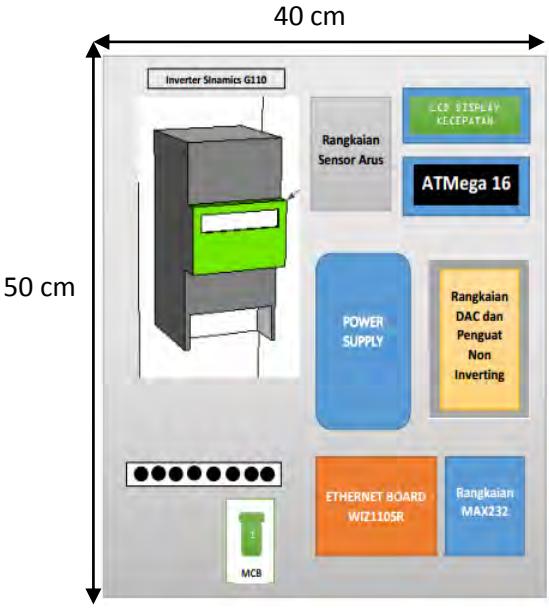
Untuk lebih jelasnya *wiring* antara PIN LCD 2x16 dan PIN ATmega 16 dapat dilihat seperti berikut

1. VSS = GND
2. VDD = +5V
3. VO = pengatur tingkat pencahayaan di hub dgn resistor 1k ke GND
4. RS = *PORT*C.0
5. RW = GND
6. E = *PORT*C.1
7. D4 = *PORT*C.2
8. D5 = *PORT*C.3
9. D6 = *PORT*C.4
10. D7 = *PORT*C.5
11. A = +5V (*power* untuk lampu *backlight* LCD)
12. K = GND (gnd untuk lampu *backlight* LCD)

3.2.6 Perancangan *Panel Box*

Perancangan *panel box* dimaksudkan sebagai tempat untuk peletakkan semua rangkaian kelistrikan yang diperlukan untuk mempermudah memberikan sumber tenaga listrik. Perancangan panel dibuat agar rangkaian – rangkaian yang diperlukan bisa lebih praktis dan ada pada satu tempat yang sama sehingga pengecekan dapat dilakukan dengan mudah dan *wiring* dapat tertata dengan rapi sehingga lebih aman dan terkendali. Panel listrik terdiri dari rangkaian *DAC* dan penguat *non inverting*, *Inverter SINAMIICS G110*, Rangkaian *Display LCD* 2X16, rangkaian Kontrol Mikrokontroler ATMega16, rangkaian *power supply* rangkaian MAX232, rangkaian sensor arus dan Modul *Ethernet WIZ110SR*.

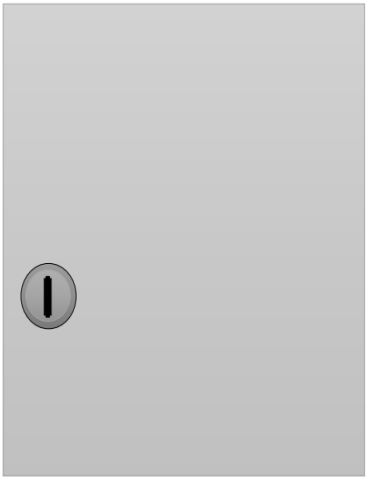
Untuk peletakkan komponen-komponen di dalam panel listrik sistem pengkabelan harus ditata secara rapi untuk memenuhi standar keamanan. Apabila ada kabel yang berserabut ujung kabel harus diberi pengaman berupa *skun* kabel untuk menghindari terjadinya hubung singkat antar komponen kelistrikan. Gambar panel kelistrikan bagian dalam dan bagian luar dapat dilihat pada Gambar 3.12 dan Gambar 3.13.



Gambar 3.12 Panel Kelistrikan Bagian Dalam

Komponen yang terdapat pada *Panel Box* kelistrikan bagian dalam :

1. Mikrokontroler ATMega16
   Sebagai kendali kontrol seluruh sistem kontrol dan monitoring kecepatan motor induksi.
2. Rangkaian *DAC*
   Untuk pengkonversi kode-kode biner dari mikrokontroler menjadi tegangan *analog* yang menuju ke *Inverter* yang nantinya akan menjadi dasar dalam pengaturan kecepatan putar motor.
3. *Inverter Sinamics* G110
   Sebagai Kendali kecepatan motor Induksi yang *input*annya dikendalikan oleh mikrokontroler sesuai dengan kecepatan putar motor yang diinginkan.
4. *Ethernet Board*
   Sebagai media komunikasi antara Mikrokontroler Atmega16 dengan *software LabVIEW* pada komputer operator
5. *Display LCD*
   Rangkaian untuk menampilkan berapa kecepatan putar motor selain pada tampilan *interface LabVIEW* pada komputer.
6. Rangkaian *Power Supply*
   Untuk men*supply* tegangan pada mikrokontroler dan perangkat kelistrikan lainnya yang membutuhkan sumber Vdc.
7. Terminal
   Sebagai penghubung kabel-kabel komponen elektronik agar terlihat lebih rapi.
8. MCB 1 Phasa
   Digunakan untuk menyambungkan sumber tegangan AC 220 Volt dari jala-jala PLN menuju rangkaian dalam panel box yang membutuhkan tegangan AC PLN, yaitu inverter sinamics dan power supply.
9. Rangkaian Sensor Arus
   Digunakan untuk mengukur besarnya arus dari *autotrafo* yang masuk ke pengereman untuk kemudian akan ditampilkan pada *LabVIEW* sebagai indikator level beban pengereman.



Gambar 3.13 Panel Kelistrikan Bagian Luar

3.3 Perancangan Perangkat Lunak (*Software)*

Pada perancangan perangkat lunak atau *software* ini akan dibahas mengenai Inisialisasi *Inverter* Sinamics G110, Inisialisasi *Port* LCD 2x16.

3.3.1 Inisialisasi *Inverter* Sinamics G110

Sebelum menggunakan *Inverter Sinamics G110* ada prosedur yang harus dilakukan yakni melakukan konfigurasi *Parameter List* pada *Inverter. Quick Comissioning* adalah cara mudah untuk mengkonfigurasi *sinamics G110* secara optimal terhadap motor tertentu. Namun ada beberapa data atau parameter yang harus dimasukkan atau diubah sesuai *name plate* pada motor yang akan digunakan, seperti tegangan *input*an motor, batas frekuensi operasi, waktu *ramp-up*, waktu *ramp-down*, dan lain-lain.

Agar nantinya motor bisa dikendalikan melalui *inverter*, maka perlu dilakukan serangkaian proses *Quick Commissioning* untuk memasukan nilai-nilai parameter. Nilai parameter yang dimasukkan harus sesuai dengan spesifikasi mekanis dari motor yang akan diatur kecepatannya agar nantinya *inverter* dan motor dapat terkoneksi dengan baik. Keterangan *name plate* motor induksi yang akan digunakan dapat dilihat pada Tabel 3.2.

Tabel 3.2 *Name Plate* Motor [3]

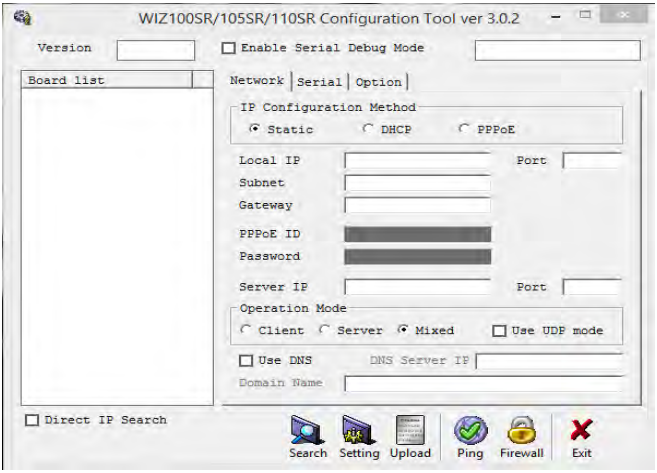
Alliance – Italy		*IEC 34 – CE*
TYPE AY 638 – 4		No. 02030688
0,18 KW	0,25 HP	1,07/0,62 A
220/380 V	1310 r/min	*LW* 52 dB (A)
CON Δ/*Y*	*Port Grade. IP* 55	50 Hz 4,7 Kg
JB/78680, 1-1998	*Work Rule* 51	*INS. CLASS F DATE* 02

Langkah-langkah pengaturan parameter *quick comissioning* untuk *Inverter Sinamics G110* lebih lengkapnya dapat dilihat pada Lampiran D.

3.3.2 *Setting IP* Rangkaian *Ethernet* WIZ110SR

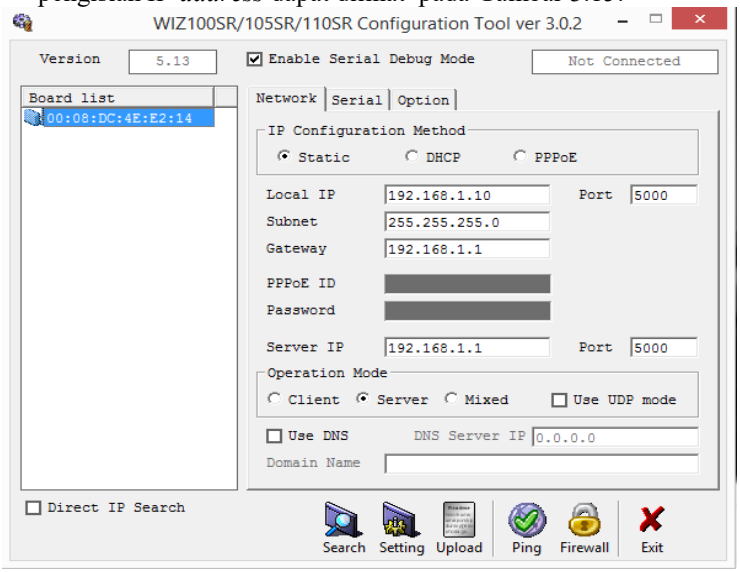
Untuk dapat mengakses Ethernet WIZ110 ke PC kita harus terlebih dahulu melakukan penyettingan *IP address*. Langkah-langkah untuk penyettingan pada Ethernet WIZ110SR adalah sebagai berikut :

1. Koneksikan terlebih dahulu antara PC dan Rangkaian Ethernet WIZ110SR dengan menggunakan kabel RJ45.
2. Pertama kita harus menginstal terlebih dahulu software WIZ10XSR *Configuration Tool*.
3. Buka software WIZ10XSR *Configuration Tool* sehingga muncul jendela seperti pada Gambar 3.14.



Gambar 3.14 Tampilan WIZ10XSR *Configuration Tool*

4. Lalu isikan Serial *IP Address* yang diinginkan, pastikan *IP address* yang diisikan sesuai agar dapat terkoneksi. Untuk pengisian *IP address* dapat dilihat pada Gambar 3.15.



Gambar 3.15 Pengisian *IP Address Ethernet WIZ110SR*

5. Setelah itu simpan *setting* IP yang telah diisikan.

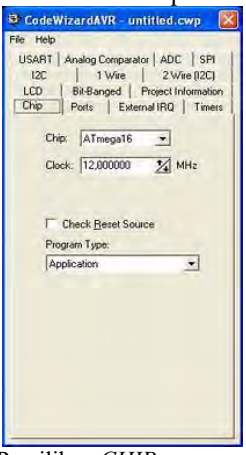
3.3.3 Inisialisasi *Port* LCD 2x16

Tampilan LCD digunakan untuk menampilkan kecepatan putar motor selain pada tampilan *interface LabVIEW* pada komputer operator, untuk membuat Inisialisasi *Port* dan membuat program yang dimasukkan pada ATMega16 kita menggunakan *software CodeVision AVR*. Langkah-langkahnya adalah sebagai berikut :

1. Buka Program Codevision AVR, lalu klik *new file*. Sehingga muncul tampilan seperti pada Gambar 3.16. Lalu tandai bagian *project.*

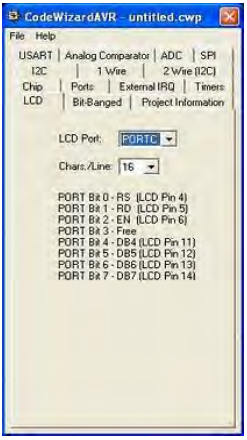
Gambar 3.16 Tampilan *Menu New Project*

2. Pada Tab *Chip*, pilih mikrokontroler seri ATMega16 dan, pada Clock setting sebesar 12 MHz. Dapat dilihat pada Gambar 3.17.



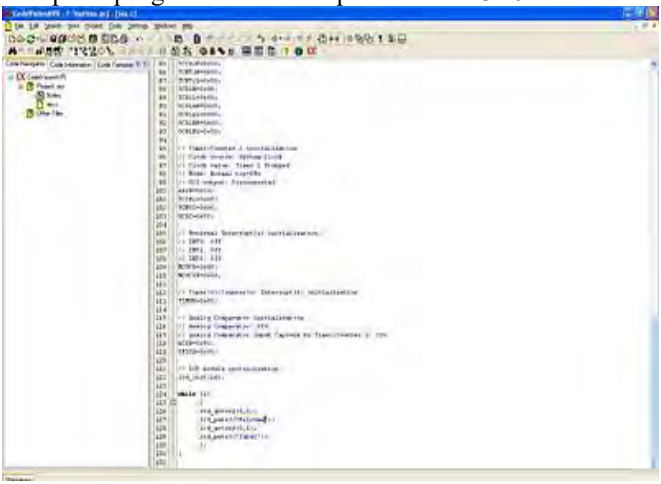
Gambar 3.17 Tampilan Pemilihan *CHIP*

3. Selanjutnnya pilih Tab LCD, pada kotak LCD *Port* pilih PORT C. Lalu akan muncul konfigurasi untuk pin C pada mikrokontroler ATMega16 seperti yang dapat dilihat pada Gambar 3.18.



Gambar 3.18 Tampilan Pemilihan *LCD Port*

4. Kemudian klik File → Generate, Pilih *Save* and *Exit*
5. Lalu beri nama file sesuai selera
6. Inisialisasi *Port* telah selesai dilakukan, maka akan muncul tampilan program otomatis seperi Gambar 3.19 dibawah ini

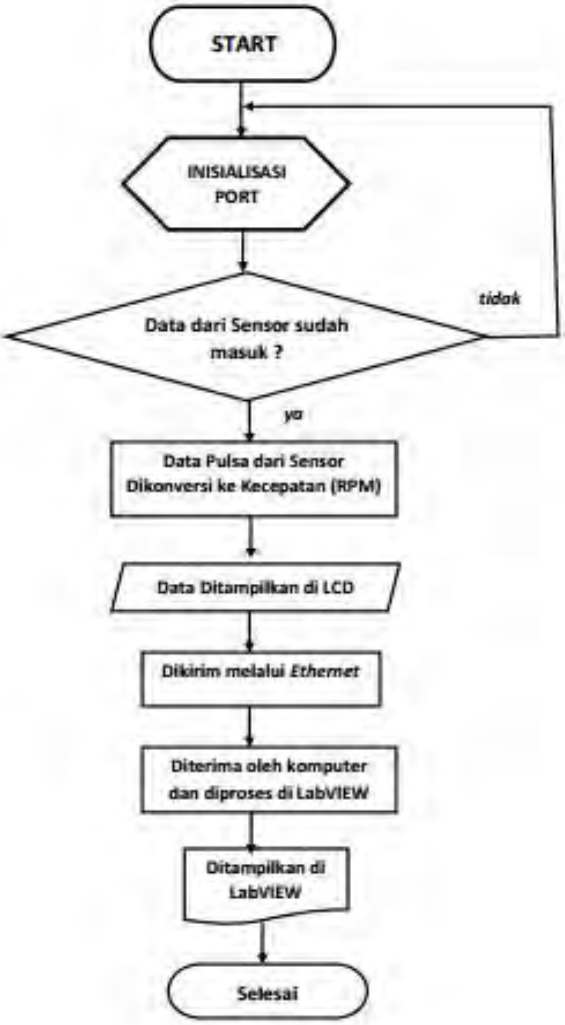


Gambar 3.19 Tampilan Pemrograman Otomatis

7. Setelah itu edit program sesuai yang diinginkan.

3.3.4 Perancangan *Software* Pembacaan Kecepatan

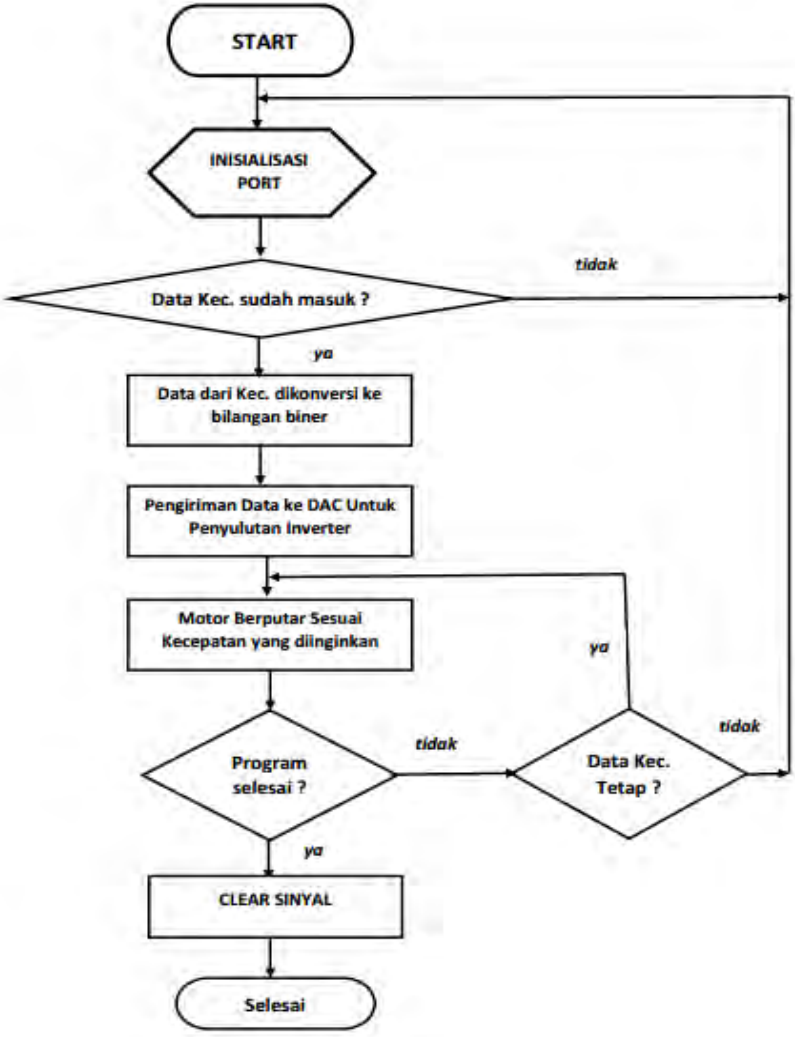
Pada Gambar 3.20 merupakan gambaran *flowchart* dari program pembacaan kecepatan yang terdapat pada mikrokontroler.



Gambar 3.20 *Flowchart* Pembacaan Kecepatan Sensor *Rotary*

3.3.5 Perancangan *Software* Pengaturan Kecepatan

Pada Gambar 3.21 merupakan gambaran *flowchart* dari program pengaturan kecepatan yang terdapat pada mikrokontroler.



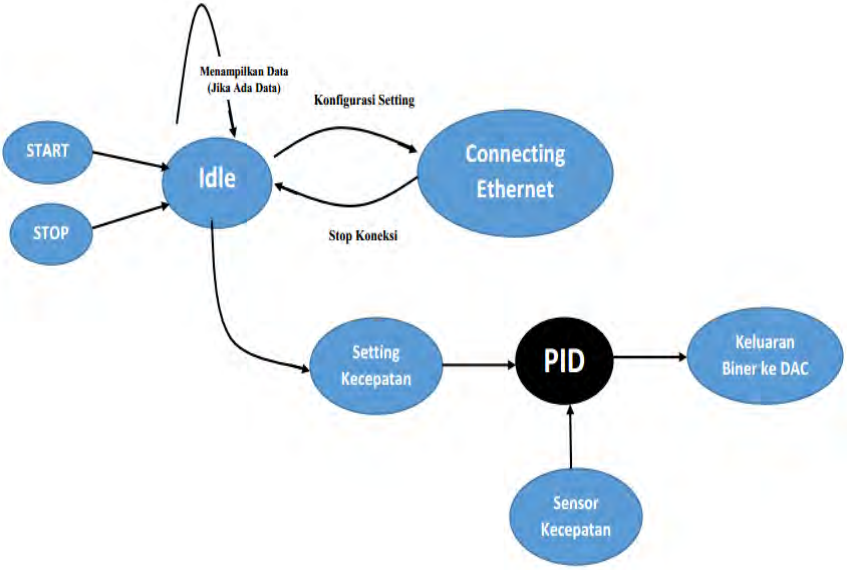
Gambar 3.21 *Flowchart* Pengaturan Kecepatan Motor

3.3.6 Perancangan *Software LabVIEW*

Software pada PC yang digunakan sebagai media *interface* adalah *software LabVIEW*. Untuk hasil yang optimal maka perlu dilakukan perancangan *software* secara matang agar *software LabVIEW* nantinya dapat difungsikan sebagaimana mestinya.

3.3.6.1 Perancangan *State Diagram*

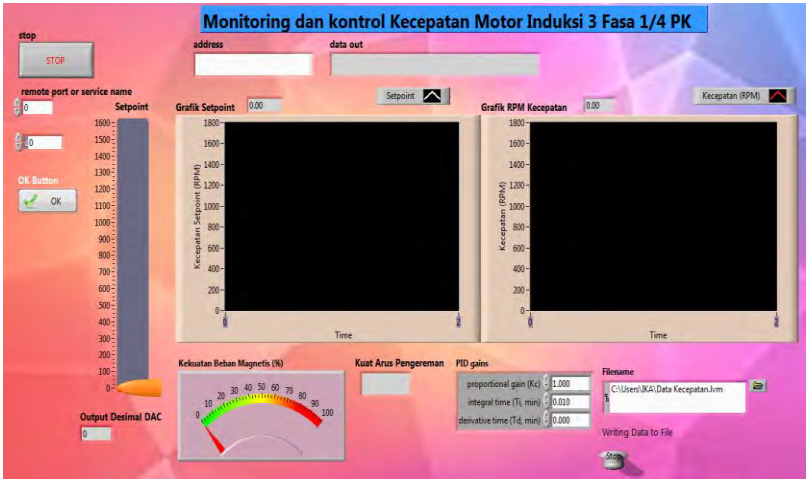
Pemodelan sistem dengan metode *state diagram* dipilih karena metode ini memudahkan dalam mendesain sistem kontrol yang kompleks.Penerapan *state diagram* pada sistem ini dilakukan dengan membuat *state* dasar yang diambil dari proses kerja setiap sub sistem. *LabVIEW* dipilih karena aplikasi *interface* ini dapat mempermudah penerapan cara kerja *state diagram* ke dalam bahasa pemrograman. Untuk *state diagram* sistem pengaturan kecepatan pada tugas akhir ini dapat dilihat pada Gambar 3.22.



Gambar 3.22 *State Diagram* Pengontrolan *PID* pada LabVIEW

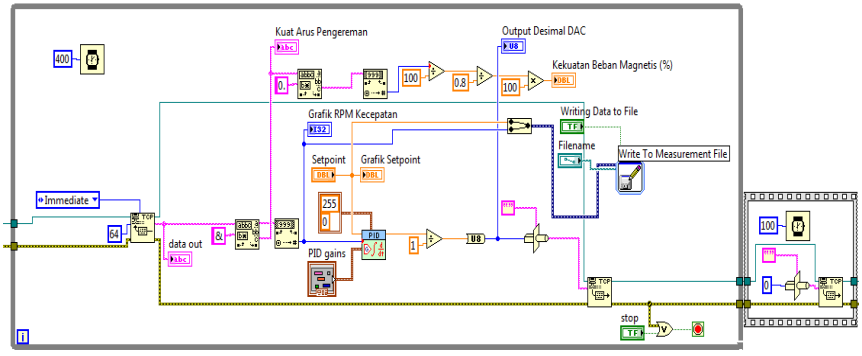
3.3.6.2 Tampilan *Software LabVIEW*

Pada Gambar 3.23 merupakan gambaran *front panel* dari perancangan *software interface* pada *LabVIEW*



Gambar 3.23 Tampilan *Front Panel* Pada *LabVIEW*

Pada Gambar 3.24 merupakan gambaran *Blok Diagram* dari perancangan *software interface* pada *LabVIEW*



Gambar 3.24 Tampilan *Blok Diagram* Pada *LabVIEW*

-----Halaman ini sengaja dikosongkan-----

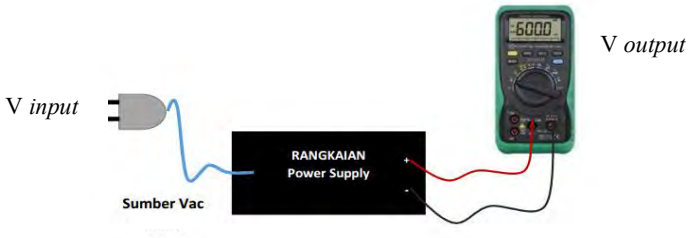
BAB IV
PENGUKURAN DAN ANALISA

Pada bab ini membahas tentang pengukuran dan analisa sistem yang telah dibuat. Pengujian sistem yang dilakukan merupakan pengujian terhadap perangkat keras dan perangkat lunak dari sistem secara keseluruhan yang telah selesai dibuat untuk memastikan agar komponen-komponen sistem yang akan digunakan dapat berfungsi dengan baik sehingga sistem nantinya akan bekerja secara optimal. Pengukuran dan analisa pada Tugas Akhir ini meliputi :

1. Pengukuran Tegangan *Output* Rangkaian *Power supply*.
2. Pengujian Tegangan Output Mikrokontroler
3. Pengukuran Rangkaian *DAC* dan Penguat *Non Inverting*
4. Pengujian Sistem Komunikasi Menggunakan WIZ110SR
5. Pengukuran Kecepatan Tanpa Sistem Kontrol Otomatis
6. Perbandingan Pengukuran Kecepatan Menggunakan Kontrol PID

4.1 Pengukuran Rangkaian *Power supply*

Pada Tugas Akhir ini perlu dilakukan pengujian rangkaian *power supply* guna mengetahui tegangan keluaran dari *power supply* apakah telah sesuai dengan tegangan *output* yang dikehendaki ataukah belum. Untuk cara pengujian kita menggunakan dua metode yaitu, pengujian *power supply* menggunakan beban dan pengujian *power supply* tidak menggunakan beban. Untuk pengukuran pertama kita melakukan pengukuran *power supply* tanpa dihubungkan ke beban. Konfigurasi pengukuran Rangkaian *Power supply* tanpa beban dapat dilihat pada Gambar 4.1 berikut ini.



Gambar 4.1 Pengukuran Tanpa Beban

Tabel 4.1 merupakan hasil dari pengujian *power supply* 5 V tanpa dihubungkan ke beban.

Tabel 4.1. Hasil Pengukuran *Power supply* 5 Volt Tanpa Beban

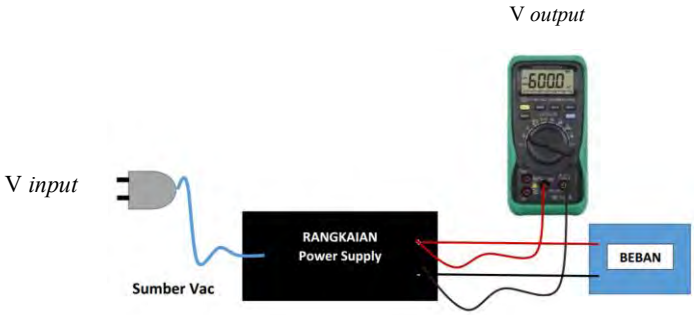
Input Power supply	***Output* (Volt)**	***% error***
222 V	5,03	0,6 %
221 V	5,01	0,2 %
219 V	4,98	0,4 %
215 V	4,96	0,8 %
210 V	4,94	1,4 %

Tabel 4.2 merupakan hasil dari pengujian *power supply* 15 V tanpa dihubungkan ke beban.

Tabel 4.2. Hasil Pengukuran *Power supply* 15 Volt Tanpa Beban

Input Power supply	***Output* (Volt)**	***% error***
223 V	15,1	0,6 %
220 V	15,01	0,06 %
219 V	14,98	0,13 %
217 V	14,93	0,46 %
210 V	14,90	0,6 %

Pengukuran selanjutnya yatu dengan menghubungkan Rangkaian *Power supply* ke beban berupa rangkaian mikrokontroler dan Sensor *Rotary Encoder* untuk *power supply* 5 V, lalu rangkaian *DAC* untuk rangkaian *power supply* 15 V kemudian diukur kembali besar tegangan *output*nya apakah mengalami *drop* tegangan atau tidak. Konfigurasi pengukuran Rangkaian *Power supply* dengan menggunakan beban dapat dilihat pada Gambar 4.2.



Gambar 4.2 Pengukuran dengan Menggunakan Beban

Tabel 4.3 merupakan hasil dari pengujian *power supply* 5 V dengan dihubungkan ke beban.

Tabel 4.3. Hasil Pengukuran *Power supply* 5 Volt dengan Beban

Input Power supply	***Output* (Volt)**	***% error***
225 V	5,00	0,0 %
221 V	4,97	0,6 %
219 V	4,94	1,2 %
215 V	4,92	1,6 %
210 V	4,90	2 %

Tabel 4.4 merupakan hasil dari pengujian *power supply* 15 V dengan dihubungkan ke beban.

Tabel 4.4. Hasil Pengukuran *Power supply* 15 Volt dengan Beban

Input Power supply	***Output* (Volt)**	***% error***
223 V	15,00	0,0 %
220 V	15,00	0,0 %
219 V	14,98	0,13 %
217 V	14,93	0,46%
210 V	14,90	0,67 %

Dari hasil pengukuran yang telah dilakukan, baik tanpa beban maupun ketika dihubungkan ke beban rangkaian *power supply* masing-masing memiliki nilai selisih *error* dengan nilai tegangan *output* yang

dikehendaki yakni sebesar 5 Volt dan 15 Volt. Selisih nilai persen *error* ini disebabkan oleh nilai tegangan *input* dari jala-jala PLN yang berubah-ubah, dan juga disebabkan masih adanya *ripple* tegangan pada tegangan *output power supply*. Namun seperti yang dapat dilihat pada tabel hasil pengukuran diatas, nilai selisih persen *error* tegangan *output* yang dihasilkan sangat kecil yakni antara hanya mencapai 2 % saja, ini berarti tegangan *output* dari *power supply* masih dalam batas aman digunakan untuk men*supply* rangkaian-rangkaian listrik yang membutuhkan tegangan VDC sebesar 5 dan 15 Volt dan tidak memberikan pengaruh yang besar pada rangkaian beban.

4.2 Pengujian Tegangan *Output* Mikrokontroler

Mikrokontroler yang digunakan dalam alat ini yaitu menggunakan ATMega 16. Untuk mengetahui apakah pin-pin pada mikrokontroler dapat digunakan dan tegangan yang dikeluarkan sesuai dengan datasheet yang ada perlu dilakukan pengukuran pada tegangan *output* mikrokontroler. Pengukuran dilakukan dengan cara mengukur tegangan pada setiap pin mikrokontroler ATMega16 yang telah dipasang pada *minimum system* untuk mengecek apakah terjadi kesalahan pada *minimum system* yang telah dibuat. Dengan mula mula program di *upload* dari laptop ke mikrokontroler ATMega16 terlebih dahulu. Program uji coba mikrokontroler dalam keadaan *High Voltage* dan *Low Voltage* dapat dilihat pada Lampiran B.2 dan B.3.

Tabel hasil pengujian PORT A untuk pin ADC pada port A0 sampai port A7 dapat dilihat pada Tabel 4.5

Tabel 4.5. Hasil Pengujian Pin ADC Mikrokontroler

Nama Port	ADC
PORT A0	0 – 1023
PORT A1	0 – 1023
PORT A2	0 – 1023
PORT A3	0 – 1023
PORT A4	0 – 1023
PORT A5	0 – 1023
PORT A6	0 – 1023
PORT A7	0 – 1023

Setelah program di *upload* maka pengukuran dilakukan pada saat tegangan dalam posisi *High Voltage* dan *Low Voltage.* Tegangan *input* yang digunakan bersumber dari *USB* laptop yaitu sebesar +5 V. Hasil pengukuran tersebut dapat dilihat pada Tabel 4.6.

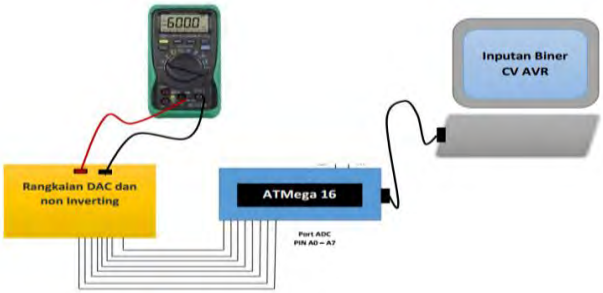
Tabel 4.6. Hasil Pengukuran Tegangan pada Setiap Pin Mikrokontroler.

Pin	***High*** **(Volt)**	***Low*** **(Volt)**	**Pin**	***High*** **(Volt)**	***Low*** **(Volt)**
B0	4,98	0	D0	4,98	0
B1	4,98	0	D1	4,98	0
B2	4,98	0	D2	4,98	0
B3	4,98	0	D3	4,98	0
B4	4,98	0	D4	4,98	0
B5	4,98	0	D5	4,98	0
B6	4,98	0	D6	4,98	0
B7	4,98	0	D7	4,98	0
C0	4,98	0	-	-	-
C1	4,98	0	-	-	-
C2	4,98	0	-	-	-
C3	4,98	0	-	-	-
C4	4,98	0	-	-	-
C5	4,98	0	-	-	-
C6	4,98	0	-	-	-
C7	4,98	0	-	-	-
Rata -Rata	**4,98**	**0**	**Rata-Rata**	**4,98**	**0**

Dilihat dari data hasil pengukuran pada Tabel 4.6, mikrokontroler (ATMega16) apabila mendapat logika 1 maka tegangan *output* sebesar 4,98 V dan saat mendapat logika 0 maka tegangan *output* sebesar 0 Volt. Ini berarti Mikrokontroler ATmega16 yang akan dipakai dalam kondisi bagus dan dapat digunakan sesuai kebutuhan.

4.3 Pengukuran Rangkaian *DAC* dan Penguat *Non Inverting*

Pada pengujian Rangkaian *DAC* kali ini dilakukan pengukuran penguat tegangan menggunakan *power supply* dengan tegangan +15 *VDC* sebagai tegangan referensi untuk rangkaian *DAC*, *input DAC* berasal dari biner 8 bit dari mikrokontroler yang dimasukkan melalui pemrograman *CV AVR* pada komputer. Bilangan biner ini kemudian akan dikonversikan oleh rangkaian *DAC* menjadi tegangan *output* berupa tegangan *analog* Selanjutnya dilakukan pengukuran tegangan *output* dari rangkaian *DAC* dengan menggunakan *multimeter*. Program untuk masukan nilai *biner* pada CV AVR dapat dilihat pada Lampiran B.5. Konfigurasi pengukuran untuk Rangkaian *DAC* dapat dilihat pada Gambar 4.3.



Gambar 4.3 Skema Pengukuran Tegangan *Output DAC*

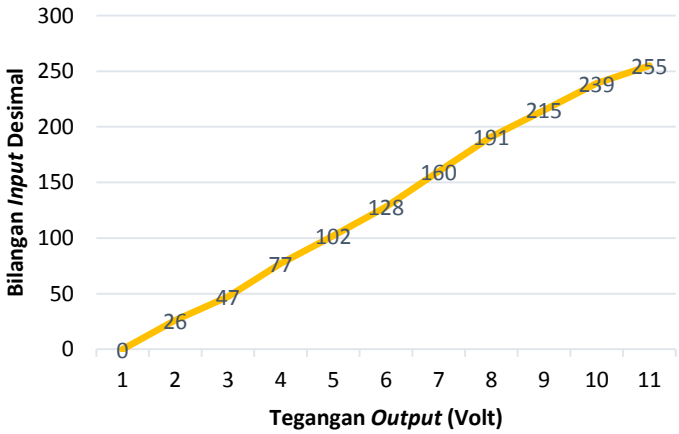
Dari Konfigurasi gambar diatas maka dapat diperoleh data pada hasil pengukuran naik dan pengukuran turun yang dapat dilihat pada Tabel 4.7.

Tabel 4.7 Hasil Pengukuran Rangkaian *DAC* dan Penguat Tegangan

Desimal	*Biner*	Tegangan *Analog 1*	Tegangan *Analog 2*
0	0000 0000	0,0063 V	0,0058 V
26	0001 1010	1,06 V	1,00 V
47	0010 1111	1,66 V	1,72 V
48	0011 0000	2,23 V	2,28 V
77	0100 1101	2,88 V	2,91 V
102	0110 0110	3,84 V	3,93 V
128	1000 0000	4,98 V	4,98 V
160	1010 0000	6,1 V	6,03 V

Desimal	*Biner*	Tegangan *Analog 1*	Tegangan *Analog 2*
191	1011 1111	6,7 V	6,82 V
215	1101 0111	7,73 V	7,85 V
239	1110 1111	8,7 V	8,87 V
255	1111 1111	9,98 V	9,94 V

Dapat dilihat pada hasil pengukuran pada Tabel 4.7 bahwa setelah dilakukan pengambilan data sebanyak 2 kali ternyata hasil tegangan *output* untuk *input* biner yang sama hasilnya mendekati ini mengindikasikan bahwa *output* rangkaian *DAC* sudah mendekati *konstan.* Untuk kenaikan nilai tegangan rangkaian dibandingkan dengan kenaikan nilai biner rangkaian *DAC* dan penguat tegangan lebih jelasnya dapat dilihat pada Gambar 4.4.



Gambar 4.4 Hasil Pengukuran Tegangan *Output DAC*

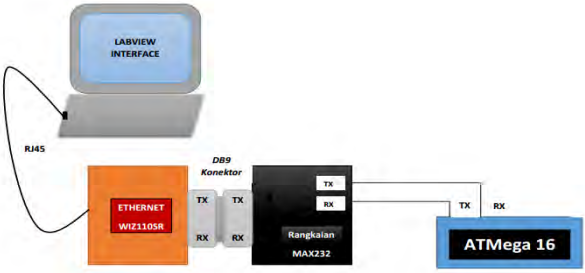
Dari Gambar 4.4 diatas dapat ditarik kesimpulan bahwa *range* tegangan *output* yang dihasilkan oleh rangkaian *DAC* dan penguat tegangan sudah sesuai dengan *spesifikasi* yang dibutuhkan oleh *inverter* yakni antara 0 – 10 Volt. Dengan hasil perhitungan resolusi per kenaikan 1 bilangan desimal sebagai berikut :

$$\text{Tegangan Resolusi} = \frac{10\ Volt}{255} = 0{,}039\ \text{Volt} = 39{,}2\ \text{mV/desimal}$$

Maka dengan demikian rangkaian *DAC* dan penguat *non inverting* tersebut aman digunakan sebagai pengkonversi tegangan digital serta penguat tegangan dari mikrokontroler *ATMega16* untuk mensuplai tegangan *input* ke *inverter* yang membutuhkan tegangan 0 sampai 10 *VDC*.

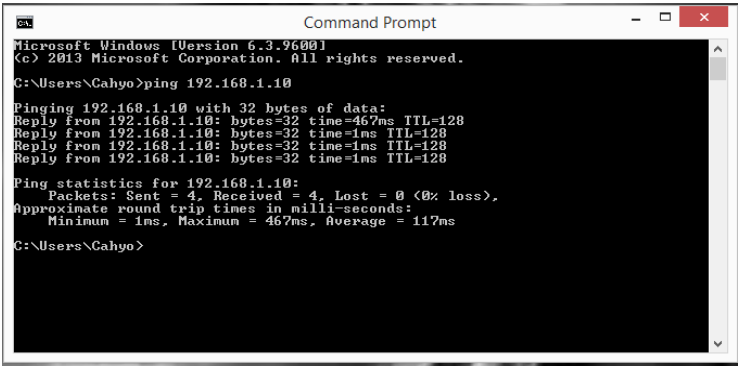
4.4 Pengujian Sistem Komunikasi Menggunakan WIZ110SR

Pada alat yang kita buat kita gunakan *serial komunikasi Ethernet yang* dihubungkan dengan *PC* Komputer dengan menggunakan kabel komunikasi serial RJ45. Rangkaian MAX232 berfunsi sebagai pengkonversi sinyal TTL dari mikrokontroler menjadi sinyal level RS232 agar bisa dibaca oleh komputer, maupun sebaliknya, sedangkan *Ethernet WIZ110SR* berfungsi sebagai media komunikasi antara mikrokontroler dengan PC komputer. Untuk *skema* rangkaian komunikasi dapat dilihat pada Gambar 4.5.



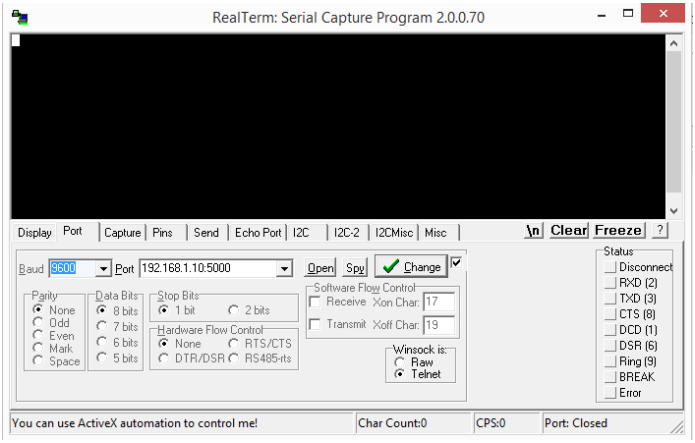
Gambar 4.5 *Skema* Rangkaian Komunikasi

Untuk mengetahui apakah antara mikrokontroler, komputer rangkaian MAX232 serta *Ethernet WIZ110SR* telah terkoneksi dengan baik maka perlu dilakukan pengujian. Untuk pengujian koneksi Untuk mengetes koneksi apakah antara *PC* dan *Ethernet* telah tersambung maka kita harus melakukan testing yaitu dengan mengirimkan "ping." Diikuti oleh *IP address Ethernet* yang telah disetting sebelumnya pada *command prompt* untuk memanggil *Ethernet*. Pengujian koneksi dapat dilihat pada Gambar 4.6.



Gambar 4.6 Pengujian Koneksi *Ethernet*

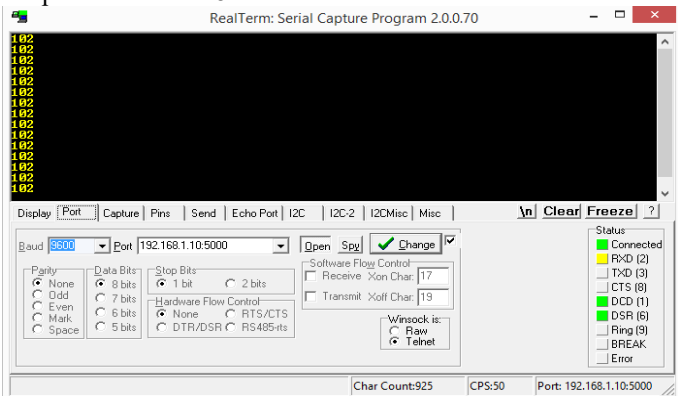
Untuk pengujian pengiriman data dari mikrokontroler ke PC dapat dilakukan dengan menggunakan *software RealTerm* pada PC. Untuk tampilan pengisian *setting*annya dapat dilihat pada Gambar 4.7.



Gambar 4.7 Konfigurasi *Setting Software RealTerm*

Setelah itu percobaan pengiriman dilakukan dengan cara melakukan pengiriman data dari mikrokontroler. Untuk Pemrograman lebih lengkapnya dapat dilihat pada Lampiran B.4. Ketika komputer telah berhasil terkoneksi dengan mirokontroler

maka komputer akan menampilkan data yang diterima dari mikrokontroler pada *software RealTerm*, untuk tampilannya dapat dilihat pada Gambar 4.8.

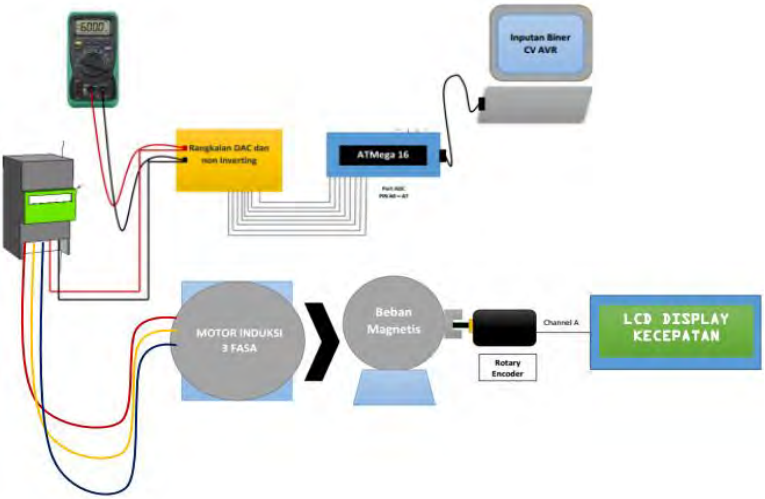


Gambar 4.8 Penerimaan Data Pada *RealTerm*

4.5 Pengukuran Kecepatan Tanpa Sistem Kontrol Otomatis

Pengukuran ini dilakukan agar dapat diketahui berapakah kecepatan motor tiga fasa saat diputar atau diberi nilai rentang tegangan antara 0 sampai dengan 10 *Volt DC*. Pengambilan data pengukuran kecepatan dilakukan beberapa kali untuk memastikan kecepatan yang dihasilkan *konstan*, mulai pengujian tanpa disambungkan ke beban pengereman magnetis maupun pengujian dengan menggunakan beban pengereman magnetis.

Pengukuran ini dilakukan dengan cara menyambungkan *plant* motor 3 fasa yang telah dikopel dengan beban rem magnetis dengan *panel box* tanpa disambungkan ke pengontrolan *PID* pada labview terlebih dahulu, jadi pengukuran kecepatan hanya ditampilkan melalui *LCD*. Hal ini dimaksudkan agar kita mendapatkan data kecepatan motor murni sebelum dilakukan sistem pengontrolan otomatis untuk nantinya dilakukan pembandingan dengan hasil pembacaan kecepatan putar motor setelah dihubungkan dengan sistem pengontrolan *PID*. Untuk konfigurasi pengambilan data kecepatan motor dapat dilihat pada Gambar 4.9.



Gambar 4.9 Konfigurasi Pengambilan Data Kecepatan Motor

4.5.1 Tanpa Beban (*Input* Pengereman Magnetis = 0 Volt)

Untuk pengukuran kecepatan motor yang pertama kita melakukan pengukuran tanpa beban, yang dimaksud disini adalah dengan tidak memberikan *input* tegangan atau *input* tegangan sama dengan 0 volt kepada beban pengereman magnetis, yang artinya beban pengereman magnetis tidak memberikan gaya kepada motor . Dimana hasil pengukuran tanpa menggunakan beban pengereman magnetis dapat dilihat pada Tabel 4.8

Tabel 4.8 Hasil Pengukuran *Plant* Motor Tanpa Beban

Pengukuran Kecepatan *Input* Pengereman Magnetis = 0 Volt							
No.	**Tegangan *Input* (Volt)**	**Frekuensi *Inverter***	**Rpm 1**	**Rpm 2**	**Rpm 3**	**Rpm Rata-rata**	**Pengukuran *Tachometer***
1	1	4,98	161	162	168	163.67	153
2	2	10,4	330	324	333	329	328
3	3	14,8	502	510	505	505.67	505
4	4	20,2	680	692	696	689.33	690

Pengukuran Kecepatan *Input* Pengereman Magnetis = 0 Volt							
No.	**Tegangan *Input* (Volt)**	**Frekuensi *Inverter***	**Rpm 1**	**Rpm 2**	**Rpm 3**	**Rpm Rata-rata**	**Pengukuran *Tachometer***
5	5	24,8	862	867	871	866.67	898
6	6	29,8	1034	1029	1031	1031.33	1077
7	7	35,4	1166	1170	1160	1165.33	1190
8	8	40	1334	1323	1318	1325	1392
9	9	44,7	1460	1470	1490	1473.33	1473,3
10	10	50	1600	1615	1615	1610	1551,67

4.5.2 Beban I (*Input* Pengereman Magnetis = 70 Volt)

Setelah melakukan pengukuran tanpa beban dan tidak terjadi masalah maka langkah selanjutnya adalah melakukan pengukuran yang kedua yaitu kita melakukan pengukuran dengan memberikan *input* pada beban rem magnetis sebesar 70 Volt.. Dimana hasil pengukuran dengan pengereman magnetis Beban I dapat dilihat pada Tabel 4.9.

Tabel 4.9 Hasil Pengukuran *Plant* Motor dengan Beban I 70 Volt

Pengukuran Kecepatan *Input* Pengereman Magnetis = 70 Volt							
No.	**Tegangan *Input* (Volt)**	**Frekuensi *Inverter***	**Rpm 1**	**Rpm 2**	**Rpm 3**	**Rpm Rata-rata**	**Pengukuran *Tachometer***
1	1	5	146	147	151	148	140
2	2	10	320	321	324	321,7	312
3	3	15	486	493	500	493	510
4	4	20	679	682	700	687	690
5	5	25	834	840	845	839,7	880
6	6	30	1007	1001	1005	1004,3	1051
7	7	35	1139	1132	1140	1137	1230
8	8	40	1299	1300	1291	1296,6	1320
9	9	45	1392	1409	1395	1398,6	1445
10	10	50	1540	1550	1545	1545	1535

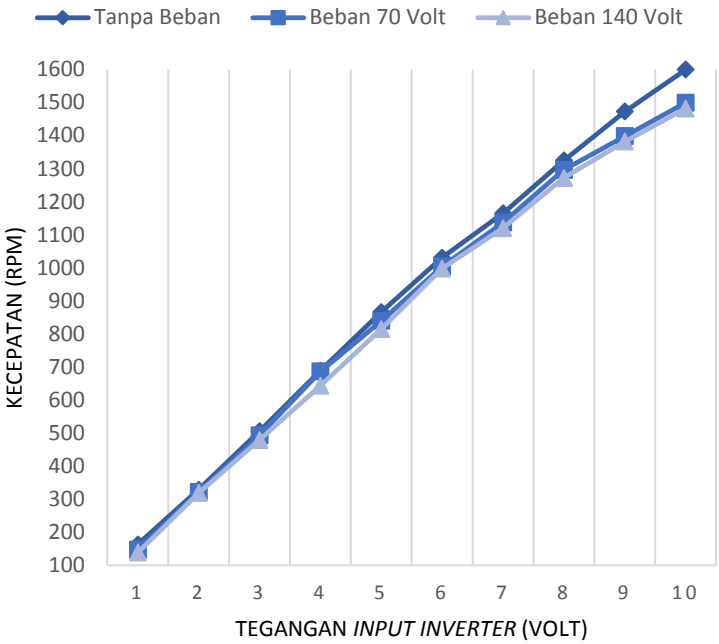
4.5.3 Beban II (*Input* Pengereman Magnetis = 140 Volt)

Untuk pengambilan data kecepatan yang terakhir yaitu dengan memberikan beban magnetis dengan *input* maksimal dari a*utotrafo* yaitu sebesar 140 Volt. Dimana hasil pengukuran dengan pengereman magnetis Beban II dapat dilihat pada Tabel 4.10.

Tabel 4.10 Hasil Pengukuran *Plant* Motor dengan Beban II 140 Volt

Pengukuran Kecepatan *Input* Pengereman Magnetis = 140 Volt							
No.	**Tegangan *Input* (Volt)**	**Frekuensi *Inverter***	**Rpm 1**	**Rpm 2**	**Rpm 3**	**Rpm Rata-rata**	**Pengukuran *Tachometer***
1	1	4,98	140	135	142	139	130
2	2	10,4	315	320	322	319	320
3	3	14,8	476	480	482	479.33	480
4	4	20,2	640	640	652	644	650
5	5	24,8	815	810	820	815	820
6	6	29,8	995	1001	998	998	990
7	7	35,4	1125	1115	1120	1120	1110
8	8	40	1274	1270	1272	1272	1265
9	9	44,7	1380	1382	1385	1382.333	1375
10	10	50	1520	1515	1525	1520	1500

Perbandingan kecepatan motor sebelum dan sesudah diberikan beban dapat dilihat pada Gambar 4.10.



Gambar 4.10 Hasil Perbandingan Kecepatan Motor

Pada Gambar 4.10 dapat dilihat bahwa motor mengalami kenaikan kecepatan secara stabil pada saat kecepatan 300 rpm hingga 1400 rpm. Untuk itu pada saat percobaan *trial and error PID* serta pada saat pengambilan data respon motor kita tentukan *step response* kecepatan 300 hingga 1400 rpm.

4.6 Metode *Trial and Error PID*

Pengambilan data ini difungsikan untuk mengetahui apa respon *plant* ketika diberikan sistem kontrol otomatis menggunakan PID sehingga nantinya dapat dilakukan perbandingan pengambilan data sebelum dan sesudah dilakukan sistem pengontrolan secara otomatis. Untuk metode *trial and error* pertama kita menentuka nilai Kp terlebih dahulu dengan mengambil 3 data sebagai perbandingan setelah itu diambil nilai Kp dengan respon yang paling baik, setelah itu dilanjutkan

dengan pengambilan data Ki dan Kd juga dilakukan dengan 3 data untuk perbandingan.

Untuk penentuan waktu *Settling time* dan *Rise time,* Grafik RPM Kecepatan pada LabVIEW dikonversi dalam bentuk data Ms. Excel terlebih dahulu untuk mengetahi *step response* waktunya, lalu kemudian dari data *step response* kita dapat menentukan *range* waktu *settling time* dan *rise time* sesuai dengan ketentuan yang telah dijelaskan sebelumnya.

4.6.1 Pengujian Respon Motor Terhadap Kontrol P

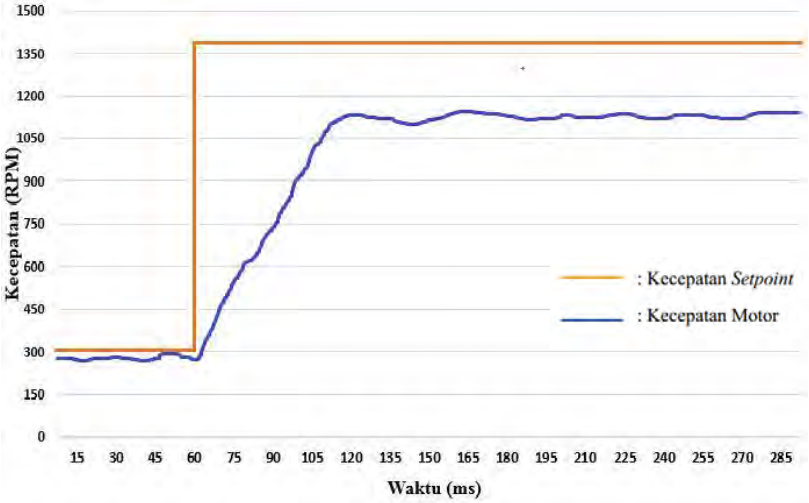
Pada uji coba ini dilakukan perhitungan *konstanta response* dari hasil pengaturan kecepatan konfigurasi *Loop* Tertutup menggunakan kontroler otomatis untuk *set point* awal kita tentukan kecepatan 300 rpm lalu kita naikkan menjadi 1400 rpm. Pengujian ini dilakukan untuk melihat respon motor dalam mencapai nilai kecepatan referensi atau *setpoint*.

Untuk metode *trial and error* yang pertama, pengambilan data dilakukan dengan mengubah nilai Kp dan mengamati respon *plant* yang terjadi. Data hasil perhitungan konstanta *response plant* terhadap perubahan nilai Kp dapat dilihat pada Tabel 4.11.

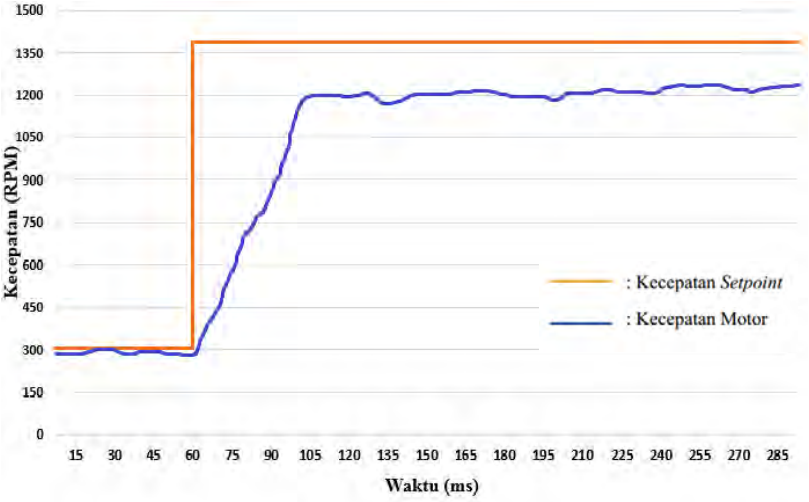
Tabel 4.11 Hasil Pengukuran *Response Plant* Terhadap Perubahan Nilai Kp

Kp	Ki	Kd	*Steady State*	*Rise Time* (detik)	*Settling Time* (detik)
2,0	0	0	1150	6,5	7,5
2,3	0	0	1200	6,2	7,7
2,7	0	0	1260	5,4	8

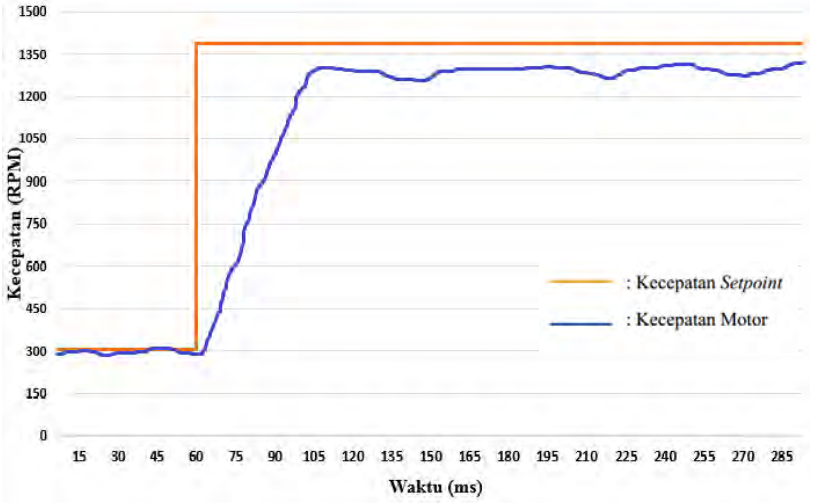
Dari data pada Tabel 4.11 menunjukkan bahwa semakin tinggi nilai Kp yang diberikan mengakibatkan kecepatan *steady state* yang dapat dicapai motor juga semakin tinggi dan mendekati *set point*. Nilai parameter Dari penentuan nilai Kp berdasarkan metode *trial and error* diatas dapat disimpulkan grafik saat Kp bernilai 2,7 motor memiliki *steady state* yang lebih tinggi dibandingkan dengan nilai Kp 2 dan Kp 2,3. Maka nilai Kp 2,7 digunakan sebagai kontrol proposional dalam sistem PID. Hal ini dikarenakan kontrol proposional berfungsi untuk memperkuat sinyal sehingga mempercepat respon mencapai *set point*. Untuk grafik *step response plant* terhadap perubahan nilai Kp dapat dilihat pada Gambar 4.11 sampai Gambar 4.13.



Gambar 4.11 Respon Motor pada Nilai Kp 2,0



Gambar 4.12 Respon Motor pada Nilai Kp 2,3



Gambar 4.13 Respon Motor pada Nilai Kp 2,7

4.6.2 Pengujian Respon Motor Terhadap Kontrol P dan I

Pada uji coba ini dilakukan perhitungan *konstanta response* dari hasil pengaturan kecepatan konfigurasi *Loop* Tertutup menggunakan kontroler otomatis untuk *set point* awal kita tentukan kecepatan 300 rpm lalu kita naikkan menjadi 1400 rpm. Pengujian ini dilakukan untuk melihat respon motor dalam mencapai nilai kecepatan referensi atau *setpoint*.

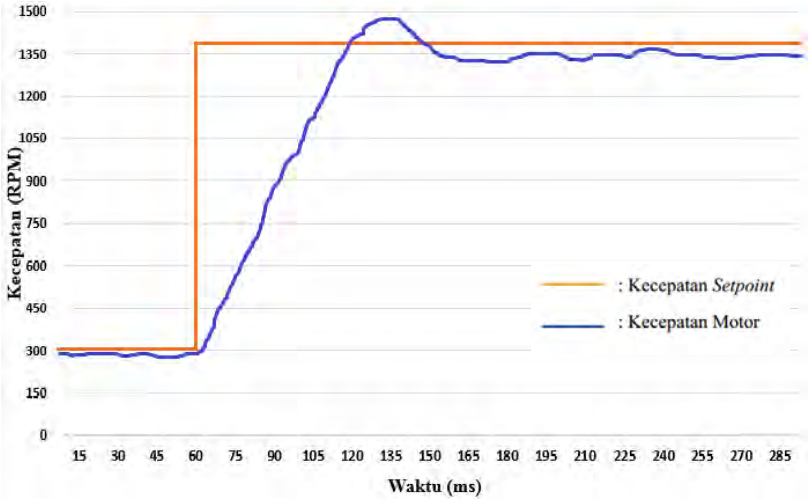
Setelah mendapatkan nilai Kp yang stabil untuk kecepatan 1400 rpm yakni 2,7 maka dilanjutkan mencari nilai Ki saat nilai Kp 2,7. Data hasil pengujian nilai Ki dengan nilai Kp 2,7 dapat dilihat pada Tabel 4.12.

Tabel 4.12 Hasil Pengukuran *Response Plant* Terhadap Perubahan Nilai Ki

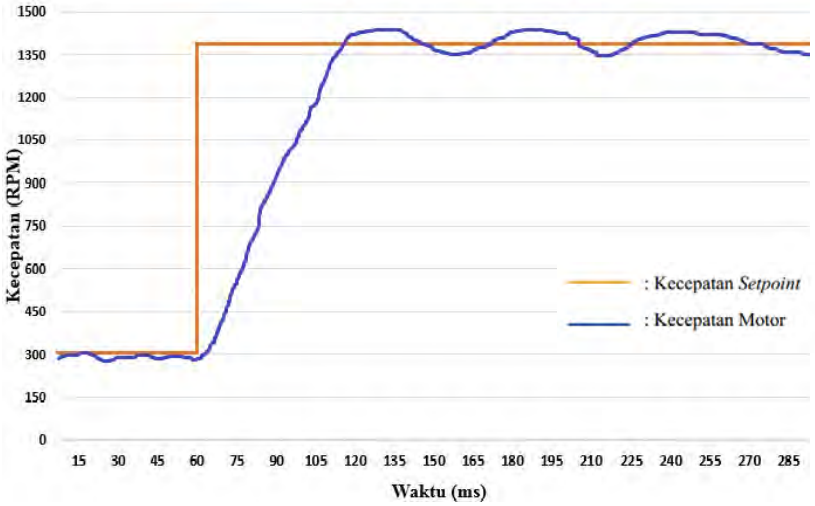
Kp	Ki	Kd	*Steady State*	*Rise Time* (detik)	*Settling Time* (detik)
2,7	1	0	1350	7,5	16,1
2,7	1,7	0	1400	7,1	12,1
2,7	2,0	0	1380	7,8	12,3

Nilai parameter Dari penentuan nilai Ki berdasarkan metode *trial and error* diatas dapat disimpulkan pada Tabel 4.12 menunjukkan bahwa saat Ki semakin besar bukan berarti respon motor semakin cepat, karena ketika nilai Ki dinaikkan menjadi 2,2 respon waktu kembali turun, untuk itu nilai Ki 1,7 digunakan sebagai kontrol integral dalam sistem PID dengan nilai *rise time* dan *settling time* Ki 1,7 lebih kecil daripada Ki 1 dan Ki 2,0.

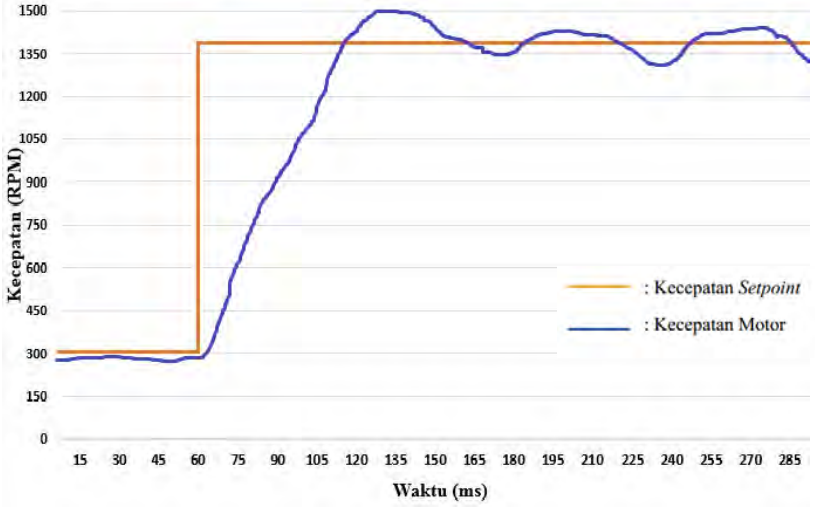
Untuk grafik *step response plant* terhadap perubahan nilai Ki dapat dilihat pada Gambar 4.14 sampai Gambar 4.16.



Gambar 4.14 Respon Motor pada Nilai Ki 1,0



Gambar 4.15 Respon Motor pada Nilai Ki 1,7



Gambar 4.16 Respon Motor pada Nilai Ki 2,0

4.6.3 Pengujian Respon Motor Terhadap Kontrol P, I, dan D

Pada uji coba ini dilakukan perhitungan *konstanta response* dari hasil pengaturan kecepatan konfigurasi *Loop* Tertutup menggunakan kontroler otomatis untuk *set point* awal kita tentukan kecepatan 300 rpm lalu kita naikkan menjadi 1400 rpm. Pengujian ini dilakukan untuk melihat respon motor dalam mencapai nilai kecepatan referensi atau *setpoint*.

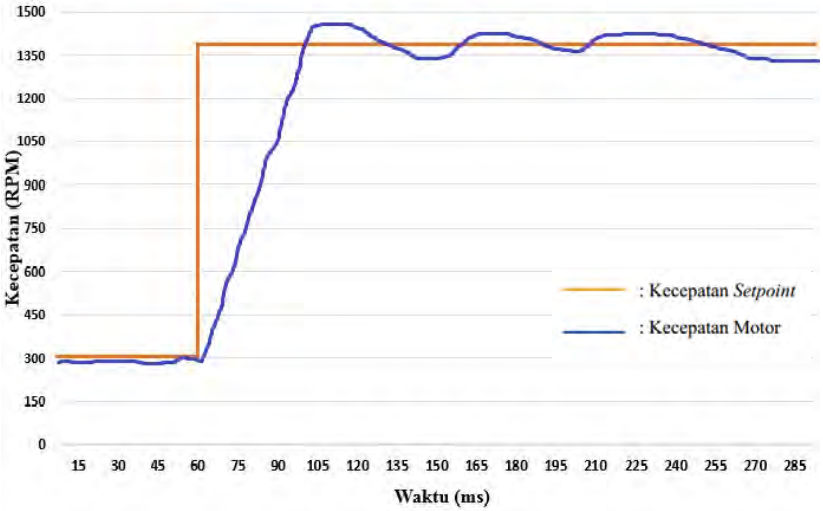
Setelah mendapatkan nilai Ki yang stabil untuk kecepatan 1400 rpm yakni 1,7 maka dilanjutkan mencari nilai Kd saat nilai Kp 2,7 dan Ki 1,7. Data hasil pengujian nilai Kd dengan nilai Kp 2,7 dan nilai Ki 1,7 dapat dilihat pada Tabel 4.13.

Tabel 4.13 Hasil Pengukuran *Response Plant* Terhadap Perubahan Nilai Kd

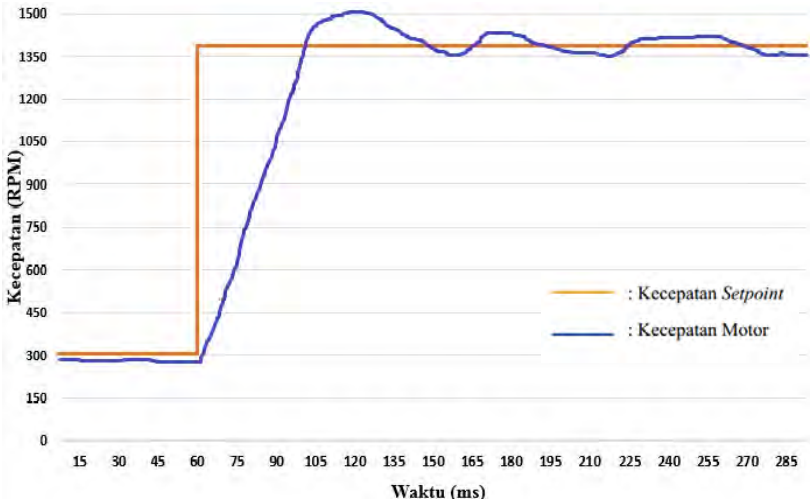
Kp	Ki	Kd	*Steady State*	*Rise Time* (detik)	*Settling Time* (detik)	*Overshoot*
2,7	1,7	0,1	1400	5,8	13,1	135
2,7	1,7	0,5	1400	5,5	17	150
2,7	1,7	1,0	1400	5,6	15,1	150

Dari data yang diperoleh pada Tabel 4.13 menunjukkan bahwa perubahan nilai Kd memiliki respon nilai *rise time* yang hampir sama namun lebih cepat dibandingkan sebelum menggunakan Kd. Maka nilai Kd 0,1 digunakan sebagai kontrol derivatif dalam sistem PID dikarenakan jika semakin besar Kd maka semakin besar nilai *settling time* dan *overshoot* yang dihasilkan, untuk itu diambil nilai Kd yang paling kecil.

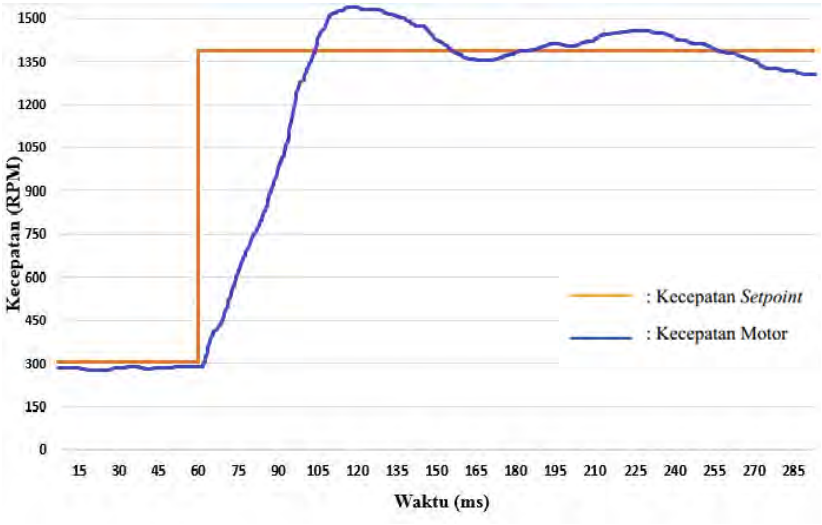
Untuk grafik *step response plant* terhadap perubahan nilai Ki dapat dilihat pada Gambar 4.17 sampai Gambar 4.19.



Gambar 4.17 Respon Motor pada Nilai Kd 0,1



Gambar 4.18 Respon Motor pada Nilai Kd 0,5



Gambar 4.19 Respon Motor pada Nilai Kd 1,0

Dari hasil pencarian nilai Kp, Ki, dan Kd dengan menggunakan metode *trial and error* diatas dapat ditarik kesimpulan nilai Kp, Ki, Kd paling *ideal* yang akan digunakan adalah sebagai berikut :

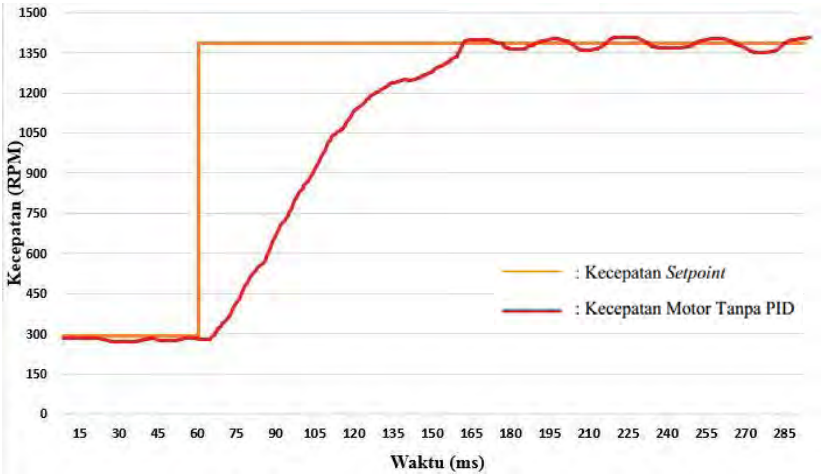
- *Kp* : 2,7
- *Ki* : 1,7
- *Kd* : 0,1

4.6.4 Pengujian Respon Motor terhadap *Rise Time* Tanpa PID

Pengujian ini dilakukan dengan melakukan perhitungan kecepatan *rise time* motor sebelum dilakukan pengontrolan otomatis dengan sistem PID. Perhitungan dilakukan dengan 3 kali yaitu perhitungan tanpa disambungkan ke beban, dan perhitungan kecepatan setelah disambungkan ke beban.

4.6.4.1 Pengujian Respon Motor Tanpa Beban

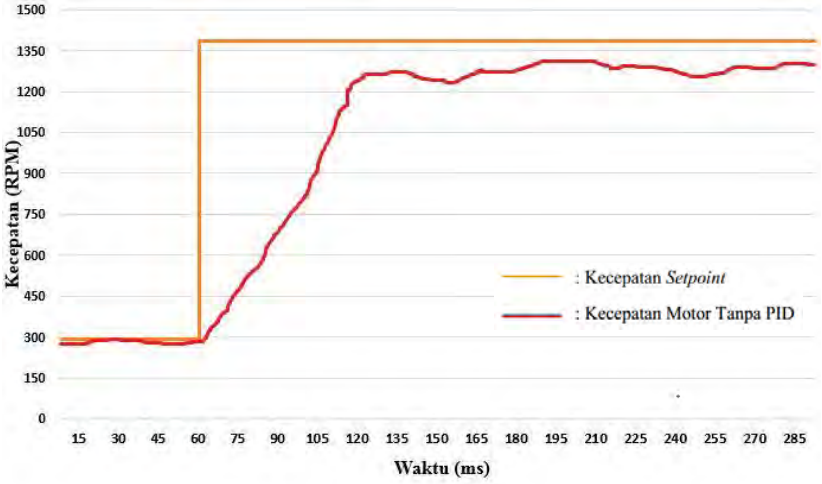
Pengambilan data yang pertama kita mengambil data hasil pengukuran kecepatan motor induksi 3 fasa tanpa dihubungkan ke beban (*input* pengereman = 0 Volt) sebelum dilakukan pengontrolan PID. Grafik hasil pembacaan kecepatan dapat dilihat pada Gambar 4.20.



Gambar 4.20 Kecepatan Motor Tanpa Beban Tanpa Kontroler PID

4.6.4.2 Pengujian Respon Motor Beban I

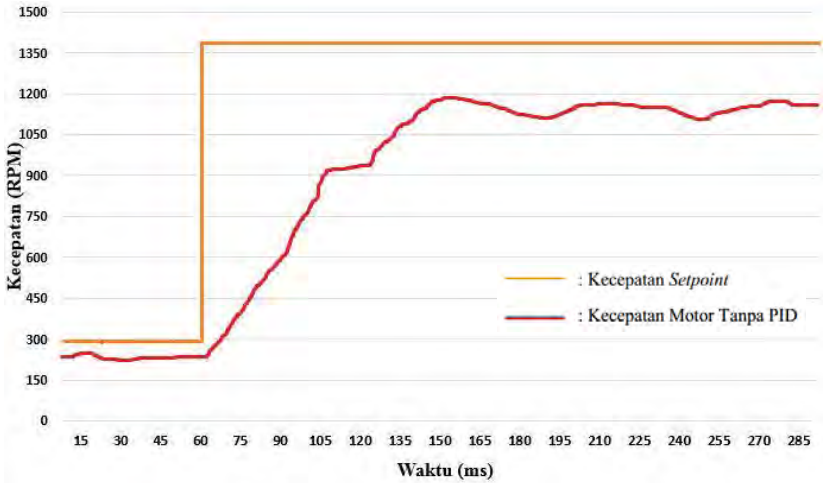
Pengambilan data yang pertama kita mengambil data hasil pengukuran kecepatan motor induksi 3 fasa dengan dihubungkan ke beban I (*input* pengereman = 70 Volt) sebelum dilakukan pengontrolan PID. Grafik hasil pembacaan kecepatan dapat dilihat pada Gambar 4.21.



Gambar 4.21 Kecepatan Motor Beban I Tanpa Kontroler PID

4.6.4.3 Pengujian Respon Motor Beban II

Pengambilan data yang pertama kita mengambil data hasil pengukuran kecepatan motor induksi 3 fasa dengan dihubungkan ke beban II (*input* pengereman = 140 Volt) sebelum dilakukan pengontrolan PID. Grafik hasil pembacaan kecepatan dapat dilihat pada Gambar 4.22.



Gambar 4.22 Kecepatan Motor Beban II Tanpa Kontroler PID

Hasil pengambilan data konstanta respon dari perhitungan kecepatan motor induksi 3 fasa tanpa kontroler PID dapat dilihat pada Tabel 4.14.

Tabel 4.14 Data Kecepatan Motor Tanpa Kontroler PID

Pengereman	***Rise Time* (Tr)**	***% Error Steady State* (Ess)**	***Settling Time* (Ts)**
Tanpa Beban	8,4 detik	3 %	9,1 detik
Beban I	6,8 detik	7,14 %	7 detik
Beban II	9,6 detik	10,28 %	11 detik

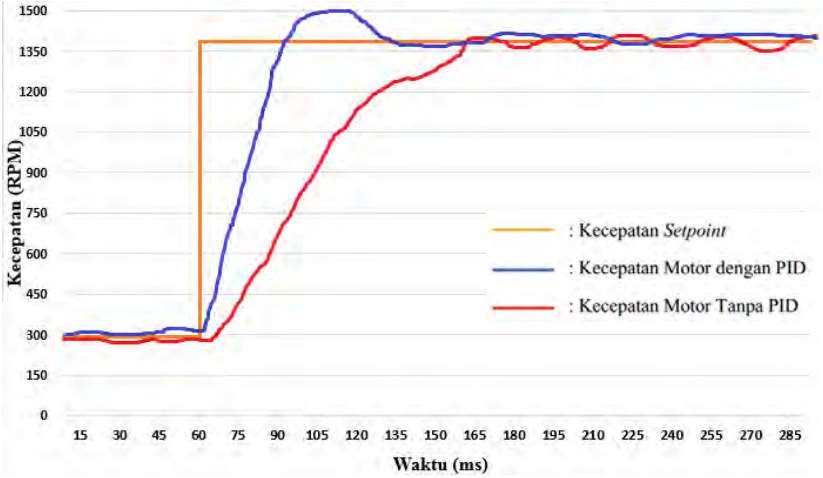
4.6.5 Pengujian Respon Motor terhadap *Rise Time* dengan PID

Pengujian ini dilakukan dengan melakukan perhitungan kecepatan *rise time* motor setelah dilakukan pengontrolan otomatis dengan sistem PID dan dilakukan perbandingan dengan sebelum

menggunakan kontrol PID. Perhitungan dilakukan dengan 3 kali yaitu perhitungan tanpa disambungkan ke beban, dan perhitungan kecepatan setelah disambungkan ke beban.

4.6.5.1 Pengujian Respon Motor Tanpa Beban

Pengujian ini dilakukan untuk melihat dan membandingkan respon motor dalam keadaan tanpa beban dalam mencapai nilai kecepatan *setpoint* sebelum dilakukan pengontrolan otomatis dengan sistem PID dan setelah menggunakan sistem PID. Pengambilan data dilakukan dengan memasukkan nilai *set point* dari 300 rpm menadi 1400 rpm dan mencatat berapa waktu *rise time* yang diperlukan *plant* untuk bisa mencapai kecepatan *setpoint.* Data hasil pengujian nilai *rise time* respon motor tanpa beban dapat dilihat pada Gambar 4.23 dan Tabel 4.15.



Gambar 4.23 Perbandingan *Step Respone* Motor Tanpa Beban

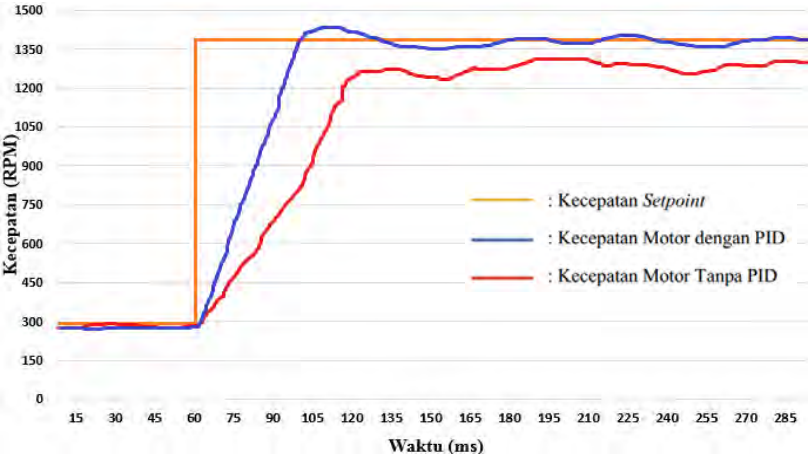
Tabel 4.15 Pengujian *Rise Time* Respon Motor Tanpa Beban

	Rise Time	***Settling time***	***Overshoot***	***% error steady state***
Tanpa PID	8,4 detik	9,1 detik	-	3 %
Dengan PID	3,1 detik	6,4 detik	150	1,68 %

Data pada Tabel 4.15 menunjukkan bahwa dengan sistem PID *rise time* dan *settling time plant* menjadi lebih cepat, *rise time* motor yang awalnya mencapai 8,4 detik turun menjadi hanya 3,6 detik ketika diberi kontrol PID, sedangakan waktu *settling time* yang awalnya 9,1 detik juga mengalami penurunan menjadi 6,4 detik.

4.6.5.2 Pengujian Respon Motor Beban I

Pengujian ini dilakukan untuk melihat dan membandingkan respon motor dalam keadaan Beban I dalam mencapai nilai kecepatan *setpoint* sebelum dilakukan pengontrolan otomatis dengan sistem PID dan setelah menggunakan sistem PID. Pengambilan data dilakukan dengan memasukkan nilai *set point* dari 300 rpm menadi 1400 rpm dan mencatat berapa waktu *rise time* yang diperlukan *plant* untuk bisa mencapai kecepatan *setpoint.* Data hasil pengujian nilai *rise time* respon motor tanpa beban dapat dilihat pada Gambar 4.24 dan Tabel 4.16.



Gambar 4.24 Perbandingan *Step Respone* Motor Beban I

Tabel 4.16 Pengujian *Rise Time* Respon Motor Beban I

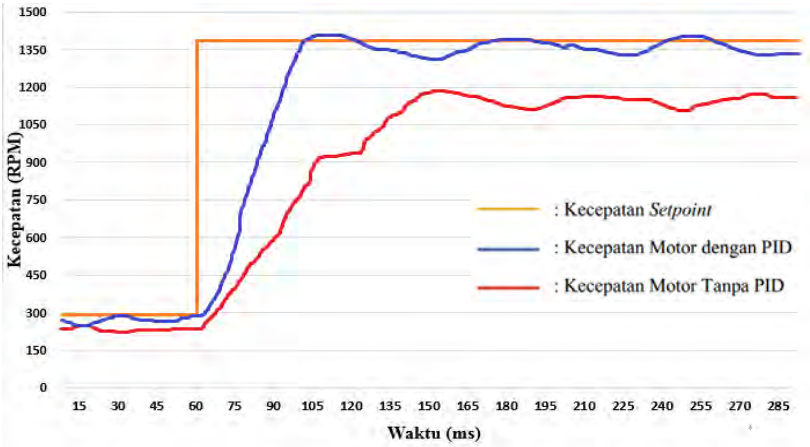
	Rise Time	***Settling time***	***Overshoot***	***% error steady state***
Tanpa PID	6,8 detik	7 detik	-	7,14 %
Dengan PID	3,6 detik	4,5 detik	50	1,3 %

Dari data yang diperoleh dari pengujian waktu *rise time plant* dengan beban I pada Tabel 4.16 dapat dilihat bahwa pembebanan pada motor dapat mempengaruhu waktu *rise time* yang diperlukan motor untuk dapat mencapai kecepatan *setpoint*, hal ini dikarenakan ketika rem magnetik diaktifkan maka pembebanan ada motor akan semakin bertambah dan mengakibatkan gerak dari motor semakin melambat maka dari itu waktu *rise time* motor juga semakin lama. Selain itu sebelum diberi kontrol PID *steady state plant* tidak mencapai *setpoint* yang ditentukan namun setelah diberi PID kecepatan putar motor dapat mencapai kecepatan *setpoint* yakni 1400 rpm.

Namun dengan sistem PID *rise time* dan *settling time plant* menjadi lebih cepat*, rise time* motor yang awalnya mencapai 6,8 detik turun menjadi hanya 3,1 detik ketika diberi kontrol PID, sedangakan waktu *settling time* yang awalnya 7 detik juga mengalami penurunan menjadi 4,5 detik. Dan kecepatan putarnya pun menjadi lebih stabil.

4.6.5.3 Pengujian Respon Motor Beban II

Pengujian ini dilakukan untuk melihat dan membandingkan respon motor dalam keadaan Beban II dalam mencapai nilai kecepatan *setpoint* sebelum dilakukan pengontrolan otomatis dengan sistem PID dan setelah menggunakan sistem PID. Pengambilan data dilakukan dengan memasukkan nilai *set point* dari 300 rpm menadi 1400 rpm dan mencatat berapa waktu *rise time* yang diperlukan *plant* untuk bisa mencapai kecepatan *setpoint.* Data hasil pengujian nilai *rise time* respon motor tanpa beban dapat dilihat pada Gambar 4.25 dan Tabel 4.17.



Gambar 4.25 Perbandingan *Step Respone* Motor Beban II

Tabel 4.17 Pengujian *Rise Time* Respon Motor Beban II

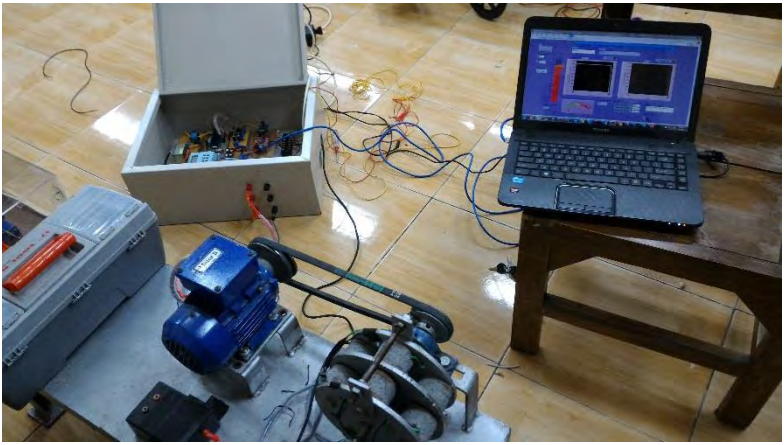
	Rise Time	***Settling time***	***Overshoot***	***% error steady state***
Tanpa PID	9,6 detik	11 detik	-	10,28 %
Dengan PID	4,9 detik	7,8 detik	15	2,14 %

Dari data yang diperoleh dari pengujian waktu *rise time plant* dengan beban II pada Tabel 4.17 dapat dilihat bahwa pembebanan pada motor dapat mempengaruhu waktu *rise time* yang diperlukan motor untuk dapat mencapai kecepatan *setpoint*, hal ini dikarenakan ketika rem magnetik diaktifkan maka pembebanan ada motor akan semakin bertambah dan mengakibatkan gerak dari motor semakin melambat maka dari itu waktu *rise time* motor juga semakin lama. Selain itu sebelum diberi kontrol PID *steady state plant* tidak mencapai *setpoint* yang ditentukan namun setelah diberi PID kecepatan putar motor dapat mencapai kecepatan *setpoint* yakni 1400 rpm.

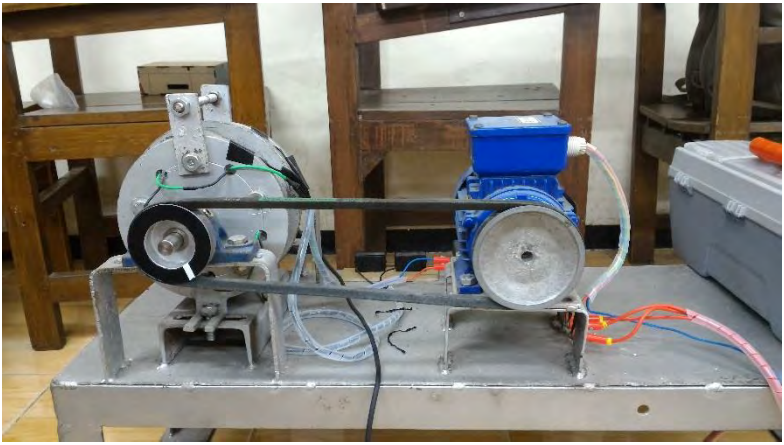
Namun dengan sistem PID *rise time* dan *settling time plant* menjadi lebih cepat, *rise time* motor yang awalnya mencapai 9,6 detik turun menjadi hanya 4,9 detik ketika diberi kontrol PID, sedangakan waktu *settling time* yang awalnya 11 detik juga mengalami penurunan menjadi 7,8 detik. Dan kecepatan putarnya pun menjadi lebih stabil.

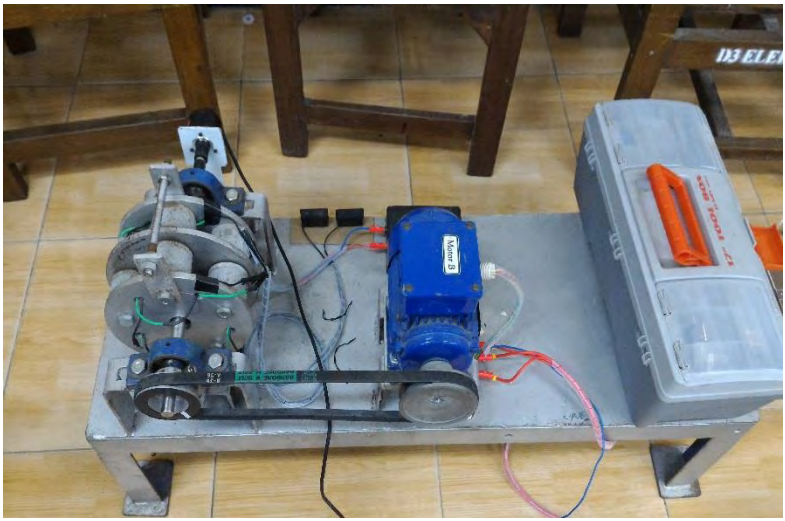
LAMPIRAN A
FOTO

Gambar Sistem Alat Keseluruhan

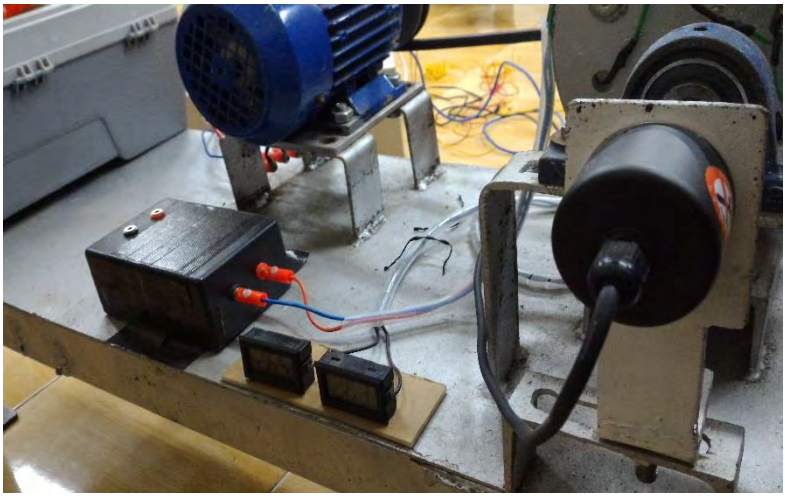


Gambar Motor Dikopel dengan Beban Pengereman





Gambar Peletakkan Sensor Temperatur dan Sensor Kecepatan



Gambar Peletakkan Komponen Pada *Panel Box*

-----Halaman ini sengaja dikosongkan-----

LAMPIRAN B
PROGRAM

B.1 Pemrograman Keseluruhan

```
/*****************************************************
This program was produced by the
CodeWizardAVR V2.05.3 Standard
Automatic Program Generator
© Copyright 1998-2011 Pavel Haiduc, HP InfoTech s.r.l.
http://www.hpinfotech.com

Project :
Version :
Date : 16/05/2016
Author : Cahyo
Company : pngping corp
Comments:


Chip type : ATmega16
Program type : Application
AVR Core Clock frequency : 11,059200 MHz
Memory model : Small
External RAM size : 0
Data Stack size : 256
*****************************************************/

int frekuensi = 0;
float pulsa;
char temp[8];

#include <string.h>
#include <mega16.h>
#include <stdlib.h>
#include <delay.h>
#include <stdio.h>

// Alphanumeric LCD functions
```

```
#include <alcd.h>
char x;
char gabung[20];
// External Interrupt 0 service routine
interrupt [EXT_INT0] void ext_int0_isr(void)
{
// Place your code here
   frekuensi++;
}

#ifndef RXB8
#define RXB8 1
#endif

#ifndef TXB8
#define TXB8 0
#endif

#ifndef UPE
#define UPE 2
#endif

#ifndef DOR
#define DOR 3
#endif

#ifndef FE
#define FE 4
#endif

#ifndef UDRE
#define UDRE 5
#endif

#ifndef RXC
#define RXC 7
#endif

#define FRAMING_ERROR (1<<FE)
```

```
#define PARITY_ERROR (1<<UPE)
#define DATA_OVERRUN (1<<DOR)
#define DATA_REGISTER_EMPTY (1<<UDRE)
#define RX_COMPLETE (1<<RXC)

// USART Receiver buffer
#define RX_BUFFER_SIZE 8
char rx_buffer[RX_BUFFER_SIZE];

#if RX_BUFFER_SIZE <= 256
unsigned char rx_wr_index,rx_rd_index,rx_counter;
#else
unsigned int rx_wr_index,rx_rd_index,rx_counter;
#endif

// This flag is set on USART Receiver buffer overflow
bit rx_buffer_overflow;

// USART Receiver interrupt service routine
interrupt [USART_RXC] void usart_rx_isr(void)
{
char status,data;
status=UCSRA;
data=UDR;
if ((status & (FRAMING_ERROR | PARITY_ERROR |
DATA_OVERRUN))==0)
   {
   rx_buffer[rx_wr_index++]=data;
#if RX_BUFFER_SIZE == 256
   // special case for receiver buffer size=256
   if (++rx_counter == 0) rx_buffer_overflow=1;
#else
   if (rx_wr_index == RX_BUFFER_SIZE) rx_wr_index=0;
   if (++rx_counter == RX_BUFFER_SIZE)
      {
      rx_counter=0;
      rx_buffer_overflow=1;
      }
#endif
```

```
   }
}

#ifndef _DEBUG_TERMINAL_IO_
// Get a character from the USART Receiver buffer
#define _ALTERNATE_GETCHAR_
#pragma used+
char getchar(void)
{
char data;
while (rx_counter==0);
data=rx_buffer[rx_rd_index++];
#if RX_BUFFER_SIZE != 256
if (rx_rd_index == RX_BUFFER_SIZE) rx_rd_index=0;
#endif
#asm("cli")
--rx_counter;
#asm("sei")
return data;
}
#pragma used-
#endif

// USART Transmitter buffer
#define TX_BUFFER_SIZE 8
char tx_buffer[TX_BUFFER_SIZE];

#if TX_BUFFER_SIZE <= 256
unsigned char tx_wr_index,tx_rd_index,tx_counter;
#else
unsigned int tx_wr_index,tx_rd_index,tx_counter;
#endif

// USART Transmitter interrupt service routine
interrupt [USART_TXC] void usart_tx_isr(void)
{
if (tx_counter)
   {
   --tx_counter;
```

```
  UDR=tx_buffer[tx_rd_index++];
#if TX_BUFFER_SIZE != 256
  if (tx_rd_index == TX_BUFFER_SIZE) tx_rd_index=0;
#endif
  }
}

#ifndef _DEBUG_TERMINAL_IO_
// Write a character to the USART Transmitter buffer
#define _ALTERNATE_PUTCHAR_
#pragma used+
void putchar(char c)
{
while (tx_counter == TX_BUFFER_SIZE);
#asm("cli")
if (tx_counter || ((UCSRA & DATA_REGISTER_EMPTY)==0))
  {
  tx_buffer[tx_wr_index++]=c;
#if TX_BUFFER_SIZE != 256
  if (tx_wr_index == TX_BUFFER_SIZE) tx_wr_index=0;
#endif
  ++tx_counter;
  }
else
  UDR=c;
#asm("sei")
}
#pragma used-
#endif

// Standard Input/Output functions
#include <stdio.h>

float Arus, Data_Arus;
char lcd_char[16];

#define ADC_VREF_TYPE 0x60

// Read the 8 most significant bits
```

```
// of the AD conversion result
unsigned char read_adc(unsigned char adc_input)
{
ADMUX=adc_input | (ADC_VREF_TYPE & 0xff);
// Delay needed for the stabilization of the ADC input voltage
delay_us(10);
// Start the AD conversion
ADCSRA|=0x40;
// Wait for the AD conversion to complete
while ((ADCSRA & 0x10)==0);
ADCSRA|=0x10;
return ADCH;
}

// Timer1 overflow interrupt service routine
interrupt [TIM1_OVF] void timer1_ovf_isr(void)
{
// Reinitialize Timer1 value
TCNT1H=0xD5CD >> 8;
TCNT1L=0xD5CD & 0xff;
// Place your code here
pulsa=(float) frekuensi*55/100;
frekuensi = 0;


}
void tampilkan_LCD()
{
   gabung[0]='\0';

   lcd_gotoxy(0,0);
   lcd_putsf("RPM");
   ftoa(pulsa,1,temp);
   lcd_gotoxy(5,0);
   lcd_puts(temp);
   Data_Arus = read_adc(0);
   Arus = ((Data_Arus*0.064811715) - 2.116171548);
   if(Data_Arus <= 35)
   {
```

```
    Arus = 0;
   }

   lcd_gotoxy(0,1);
   lcd_putsf("Arus= ");
   ftoa(Arus,2,lcd_char);
   lcd_puts(lcd_char);
   strcat(gabung,temp);
   strcat(gabung,"&");
   strcat(gabung,lcd_char);
   //delay_ms(70);
   //lcd_clear();
}

// Declare your global variables here

void main(void)
{
// Declare your local variables here


// Input/Output Ports initialization
// Port A initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTA=0x00;
DDRA=0x00;

// Port B initialization
// Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out
Func2=Out Func1=Out Func0=Out
// State7=0 State6=0 State5=0 State4=0 State3=0 State2=0 State1=0
State0=0
PORTB=0x00;
DDRB=0xFF;

// Port C initialization
```

```
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTC=0x00;
DDRC=0x00;

// Port D initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTD=0x00;
DDRD=0x00;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=0xFF
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: 10,800 kHz
// Mode: Normal top=0xFFFF
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer1 Overflow Interrupt: On
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x05;
TCNT1H=0xD5;
```

```
TCNT1L=0xCD;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer2 Stopped
// Mode: Normal top=0xFF
// OC2 output: Disconnected
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;

// External Interrupt(s) initialization
// INT0: On
// INT0 Mode: Falling Edge
// INT1: Off
// INT2: Off
GICR|=0x40;
MCUCR=0x02;
MCUCSR=0x00;
GIFR=0x40;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x04;

// USART initialization
// Communication Parameters: 8 Data, 1 Stop, No Parity
// USART Receiver: On
// USART Transmitter: On
// USART Mode: Asynchronous
// USART Baud Rate: 9600
UCSRA=0x00;
UCSRB=0xD8;
```

```
UCSRC=0x86;
UBRRH=0x00;
UBRRL=0x47;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80;
SFIOR=0x00;

// ADC initialization
// ADC Clock frequency: 691,200 kHz
// ADC Voltage Reference: AREF pin
// ADC Auto Trigger Source: Free Running
ADMUX=ADC_VREF_TYPE & 0xff;
ADCSRA=0xA4;
SFIOR&=0x1F;

// SPI initialization
// SPI disabled
SPCR=0x00;

// TWI initialization
// TWI disabled
TWCR=0x00;

// Alphanumeric LCD initialization
// Connections are specified in the
// Project|Configure|C Compiler|Libraries|Alphanumeric LCD menu:
// RS - PORTC Bit 0
// RD - PORTC Bit 1
// EN - PORTC Bit 2
// D4 - PORTC Bit 4
// D5 - PORTC Bit 5
// D6 - PORTC Bit 6
// D7 - PORTC Bit 7
// Characters/line: 16
lcd_init(16);
```

```
// Global enable interrupts
#asm("sei")

while (1)
      {
      // Place your code here
      tampilkan_LCD();
      puts(gabung);
      x = getchar();
      if (x==0){pulsa=0;};
      PORTB = x;
      delay_ms(250);
      lcd_clear();
      }
}
```

B.2 Pemrograman Pengecekan Tegangan Mikrokontroler *High Voltage*

```
/*****************************************************
This program was produced by the
CodeWizardAVR V2.05.3 Standard
Automatic Program Generator
© Copyright 1998-2011 Pavel Haiduc, HP InfoTech s.r.l.
http://www.hpinfotech.com

Project :
Version :
Date    : 25/05/2016
Author  : Cahyo
Company : pngping corp
Comments:


Chip type               : ATmega16
Program type            : Application
AVR Core Clock frequency: 12,000000 MHz
Memory model            : Small
External RAM size       : 0
Data Stack size         : 256
*****************************************************/

#include <mega16.h>
#include <string.h>
#include <stdlib.h>
#include <delay.h>
#include <stdio.h>
float data;
char lcd_char[16];
// Alphanumeric LCD functions
#include <alcd.h>

#define ADC_VREF_TYPE 0x00

// Read the AD conversion result
```

```
unsigned int read_adc(unsigned char adc_input)
{
ADMUX=adc_input | (ADC_VREF_TYPE & 0xff);
// Delay needed for the stabilization of the ADC input voltage
delay_us(10);
// Start the AD conversion
ADCSRA|=0x40;
// Wait for the AD conversion to complete
while ((ADCSRA & 0x10)==0);
ADCSRA|=0x10;
return ADCW;
}

// Declare your global variables here
void tampilkan_LCD()
{
data = read_adc(0);
//data = read_adc(1);
//data = read_adc(2);
//data = read_adc(3);
//data = read_adc(4);
//data = read_adc(5);
//data = read_adc(6);
//data = read_adc(7);
//adc = (data*1023);
lcd_gotoxy(0,1);
lcd_putsf("data= ");
ftoa(data,2,lcd_char);
lcd_puts(lcd_char);
delay_ms(100);
lcd_clear();
}
void main(void)
{
// Declare your local variables here

// Input/Output Ports initialization
// Port A initialization
```

```
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTA=0x00;
DDRA=0x00;

// Port B initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTB=0xFF;
DDRB=0x00;

// Port C initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTC=0xFF;
DDRC=0x00;

// Port D initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTD=0x00;
DDRD=0x00;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=0xFF
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;
```

```
// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer1 Stopped
// Mode: Normal top=0xFFFF
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer2 Stopped
// Mode: Normal top=0xFF
// OC2 output: Disconnected
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;

// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: Off
MCUCR=0x00;
```

```
MCUCSR=0x00;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x00;

// USART initialization
// USART disabled
UCSRB=0x00;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80;
SFIOR=0x00;

// ADC initialization
// ADC Clock frequency: 93,750 kHz
// ADC Voltage Reference: AREF pin
// ADC Auto Trigger Source: Free Running
ADMUX=ADC_VREF_TYPE & 0xff;
ADCSRA=0xA7;
SFIOR&=0x1F;

// SPI initialization
// SPI disabled
SPCR=0x00;

// TWI initialization
// TWI disabled
TWCR=0x00;

// Alphanumeric LCD initialization
// Connections are specified in the
// Project|Configure|C Compiler|Libraries|Alphanumeric LCD menu:
// RS - PORTD Bit 0
// RD - PORTD Bit 1
// EN - PORTD Bit 2
// D4 - PORTD Bit 4
// D5 - PORTD Bit 5
```

```
// D6 - PORTD Bit 6
// D7 - PORTD Bit 7
// Characters/line: 16
lcd_init(16);

while (1)
      {
      // Place your code here
       tampilkan_LCD();
       PORTB = 255;
       PORTC = 255;
      }
}
```

B.3 Pemrograman Pengecekan Tegangan Mikrokontroler *Low Voltage*

```
/*****************************************************
This program was produced by the
CodeWizardAVR V2.05.3 Standard
Automatic Program Generator
© Copyright 1998-2011 Pavel Haiduc, HP InfoTech s.r.l.
http://www.hpinfotech.com

Project :
Version :
Date    : 25/05/2016
Author  : Cahyo
Company : pngping corp
Comments:


Chip type               : ATmega16
Program type            : Application
AVR Core Clock frequency: 12,000000 MHz
Memory model            : Small
External RAM size       : 0
Data Stack size         : 256
*****************************************************/

#include <mega16.h>
#include <string.h>
#include <stdlib.h>
#include <delay.h>
#include <stdio.h>
float data;
char lcd_char[16];
// Alphanumeric LCD functions
#include <alcd.h>

#define ADC_VREF_TYPE 0x00

// Read the AD conversion result
```

```
unsigned int read_adc(unsigned char adc_input)
{
ADMUX=adc_input | (ADC_VREF_TYPE & 0xff);
// Delay needed for the stabilization of the ADC input voltage
delay_us(10);
// Start the AD conversion
ADCSRA|=0x40;
// Wait for the AD conversion to complete
while ((ADCSRA & 0x10)==0);
ADCSRA|=0x10;
return ADCW;
}

// Declare your global variables here
void tampilkan_LCD()
{
data = read_adc(0);
//data = read_adc(1);
//data = read_adc(2);
//data = read_adc(3);
//data = read_adc(4);
//data = read_adc(5);
//data = read_adc(6);
//data = read_adc(7);
//adc = (data*1023);
lcd_gotoxy(0,1);
lcd_putsf("data= ");
ftoa(data,2,lcd_char);
lcd_puts(lcd_char);
delay_ms(100);
lcd_clear();
}
void main(void)
{
// Declare your local variables here

// Input/Output Ports initialization
// Port A initialization
```

```
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTA=0x00;
DDRA=0x00;

// Port B initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTB=0xFF;
DDRB=0x00;

// Port C initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTC=0xFF;
DDRC=0x00;

// Port D initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTD=0x00;
DDRD=0x00;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=0xFF
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;
```

```
// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer1 Stopped
// Mode: Normal top=0xFFFF
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer2 Stopped
// Mode: Normal top=0xFF
// OC2 output: Disconnected
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;

// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: Off
MCUCR=0x00;
```

```
MCUCSR=0x00;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x00;

// USART initialization
// USART disabled
UCSRB=0x00;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80;
SFIOR=0x00;

// ADC initialization
// ADC Clock frequency: 93,750 kHz
// ADC Voltage Reference: AREF pin
// ADC Auto Trigger Source: Free Running
ADMUX=ADC_VREF_TYPE & 0xff;
ADCSRA=0xA7;
SFIOR&=0x1F;

// SPI initialization
// SPI disabled
SPCR=0x00;

// TWI initialization
// TWI disabled
TWCR=0x00;

// Alphanumeric LCD initialization
// Connections are specified in the
// Project|Configure|C Compiler|Libraries|Alphanumeric LCD menu:
// RS - PORTD Bit 0
// RD - PORTD Bit 1
// EN - PORTD Bit 2
// D4 - PORTD Bit 4
// D5 - PORTD Bit 5
```

```
// D6 - PORTD Bit 6
// D7 - PORTD Bit 7
// Characters/line: 16
lcd_init(16);

while (1)
      {
      // Place your code here
       tampilkan_LCD();
       PORTB = 0;
       PORTC = 0;
      }
}
```

B.4 Pemrograman Pengiriman Data Melalui *Real Term*

```
/*****************************************************
This program was produced by the
CodeWizardAVR V2.05.3 Standard
Automatic Program Generator
© Copyright 1998-2011 Pavel Haiduc, HP InfoTech s.r.l.
http://www.hpinfotech.com

Project :
Version :
Date    : 08/06/2016
Author  : Cahyo
Company : pngping corp
Comments:


Chip type               : ATmega16
Program type            : Application
AVR Core Clock frequency: 12,000000 MHz
Memory model            : Small
External RAM size       : 0
Data Stack size         : 256
*****************************************************/

#include <mega16.h>
#include <stdio.h>
#include <stdlib.h>
#include <delay.h>
 char i;
 char x [8];
// Alphanumeric LCD functions
#include <alcd.h>

#ifndef RXB8
#define RXB8 1
#endif

#ifndef TXB8
```

```
#define TXB8 0
#endif

#ifndef UPE
#define UPE 2
#endif

#ifndef DOR
#define DOR 3
#endif

#ifndef FE
#define FE 4
#endif

#ifndef UDRE
#define UDRE 5
#endif

#ifndef RXC
#define RXC 7
#endif

#define FRAMING_ERROR (1<<FE)
#define PARITY_ERROR (1<<UPE)
#define DATA_OVERRUN (1<<DOR)
#define DATA_REGISTER_EMPTY (1<<UDRE)
#define RX_COMPLETE (1<<RXC)

// USART Receiver buffer
#define RX_BUFFER_SIZE 8
char rx_buffer[RX_BUFFER_SIZE];

#if RX_BUFFER_SIZE <= 256
unsigned char rx_wr_index,rx_rd_index,rx_counter;
#else
unsigned int rx_wr_index,rx_rd_index,rx_counter;
#endif
```

```
// This flag is set on USART Receiver buffer overflow
bit rx_buffer_overflow;

// USART Receiver interrupt service routine
interrupt [USART_RXC] void usart_rx_isr(void)
{
char status,data;
status=UCSRA;
data=UDR;
if ((status & (FRAMING_ERROR | PARITY_ERROR |
DATA_OVERRUN))==0)
   {
   rx_buffer[rx_wr_index++]=data;
#if RX_BUFFER_SIZE == 256
   // special case for receiver buffer size=256
   if (++rx_counter == 0) rx_buffer_overflow=1;
#else
   if (rx_wr_index == RX_BUFFER_SIZE) rx_wr_index=0;
   if (++rx_counter == RX_BUFFER_SIZE)
      {
      rx_counter=0;
      rx_buffer_overflow=1;
      }
#endif
   }
}

#ifndef _DEBUG_TERMINAL_IO_
// Get a character from the USART Receiver buffer
#define _ALTERNATE_GETCHAR_
#pragma used+
char getchar(void)
{
char data;
while (rx_counter==0);
data=rx_buffer[rx_rd_index++];
#if RX_BUFFER_SIZE != 256
if (rx_rd_index == RX_BUFFER_SIZE) rx_rd_index=0;
#endif
```

```
#asm("cli")
--rx_counter;
#asm("sei")
return data;
}
#pragma used-
#endif

// USART Transmitter buffer
#define TX_BUFFER_SIZE 8
char tx_buffer[TX_BUFFER_SIZE];

#if TX_BUFFER_SIZE <= 256
unsigned char tx_wr_index,tx_rd_index,tx_counter;
#else
unsigned int tx_wr_index,tx_rd_index,tx_counter;
#endif

// USART Transmitter interrupt service routine
interrupt [USART_TXC] void usart_tx_isr(void)
{
if (tx_counter)
   {
   --tx_counter;
   UDR=tx_buffer[tx_rd_index++];
#if TX_BUFFER_SIZE != 256
   if (tx_rd_index == TX_BUFFER_SIZE) tx_rd_index=0;
#endif
   }
}

#ifndef _DEBUG_TERMINAL_IO_
// Write a character to the USART Transmitter buffer
#define _ALTERNATE_PUTCHAR_
#pragma used+
void putchar(char c)
{
while (tx_counter == TX_BUFFER_SIZE);
#asm("cli")
```

```
if (tx_counter || ((UCSRA & DATA_REGISTER_EMPTY)==0))
   {
   tx_buffer[tx_wr_index++]=c;
#if TX_BUFFER_SIZE != 256
   if (tx_wr_index == TX_BUFFER_SIZE) tx_wr_index=0;
#endif
   ++tx_counter;
   }
else
   UDR=c;
#asm("sei")
}
#pragma used-
#endif

// Standard Input/Output functions
#include <stdio.h>

// Declare your global variables here
void hah ()
{
lcd_gotoxy(0,0);
lcd_putsf("128");
}
void main(void)
{
// Declare your local variables here

// Input/Output Ports initialization
// Port A initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTA=0x00;
DDRA=0x00;

// Port B initialization
```

```
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTB=0x00;
DDRB=0x00;

// Port C initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTC=0x00;
DDRC=0x00;

// Port D initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTD=0x00;
DDRD=0x00;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=0xFF
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer1 Stopped
// Mode: Normal top=0xFFFF
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
```

```
// Input Capture on Falling Edge
// Timer1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer2 Stopped
// Mode: Normal top=0xFF
// OC2 output: Disconnected
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;

// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: Off
MCUCR=0x00;
MCUCSR=0x00;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x00;

// USART initialization
// Communication Parameters: 8 Data, 1 Stop, No Parity
// USART Receiver: On
```

```
// USART Transmitter: On
// USART Mode: Asynchronous
// USART Baud Rate: 9600
UCSRA=0x00;
UCSRB=0xD8;
UCSRC=0x86;
UBRRH=0x00;
UBRRL=0x4D;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80;
SFIOR=0x00;

// ADC initialization
// ADC disabled
ADCSRA=0x00;

// SPI initialization
// SPI disabled
SPCR=0x00;

// TWI initialization
// TWI disabled
TWCR=0x00;

// Alphanumeric LCD initialization
// Connections are specified in the
// Project|Configure|C Compiler|Libraries|Alphanumeric LCD menu:
// RS - PORTC Bit 0
// RD - PORTC Bit 1
// EN - PORTC Bit 2
// D4 - PORTC Bit 4
// D5 - PORTC Bit 5
// D6 - PORTC Bit 6
// D7 - PORTC Bit 7
// Characters/line: 16
lcd_init(16);
```

```
// Global enable interrupts
#asm("sei")

while (1)
      {
      // Place your code here
      hah();
      i=0; i=128 ; i++;;
      }
}
```

B.5 Pemrograman Rangkaian *DAC* dan Penguat *Non Inverting*

```
/*****************************************************
This program was produced by the
CodeWizardAVR V2.05.3 Standard
Automatic Program Generator
© Copyright 1998-2011 Pavel Haiduc, HP InfoTech s.r.l.
http://www.hpinfotech.com

Project :
Version :
Date    : 11/04/2016
Author  : Cahyo
Company : pngping corp
Comments:


Chip type               : ATmega16
Program type            : Application
AVR Core Clock frequency: 12,000000 MHz
Memory model            : Small
External RAM size       : 0
Data Stack size         : 256
*****************************************************/

#include <mega16.h>
#include <delay.h>
// Declare your global variables here

void main(void)
{
// Declare your local variables here

// Input/Output Ports initialization
// Port A initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
```

```
PORTA=0x00;
DDRA=0x00;

// Port B initialization
// Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out
Func2=Out Func1=Out Func0=Out
// State7=0 State6=0 State5=0 State4=0 State3=0 State2=0 State1=0
State0=0
PORTB=0x00;
DDRB=0xFF;

// Port C initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTC=0x00;
DDRC=0x00;

// Port D initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTD=0x00;
DDRD=0x00;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=0xFF
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer1 Stopped
```

```
// Mode: Normal top=0xFFFF
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer2 Stopped
// Mode: Normal top=0xFF
// OC2 output: Disconnected
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;

// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: Off
MCUCR=0x00;
MCUCSR=0x00;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x00;
```

```
// USART initialization
// USART disabled
UCSRB=0x00;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80;
SFIOR=0x00;

// ADC initialization
// ADC disabled
ADCSRA=0x00;

// SPI initialization
// SPI disabled
SPCR=0x00;

// TWI initialization
// TWI disabled
TWCR=0x00;

while (1)
      {
      // Place your code here
        PORTB = 0b11111111 ;
      }
}
```

LAMPIRAN C
DATASHEET

1. ***Datasheet*** **Inverter Sinamics G110**

1 Overview

1.1 The SINAMICS G110

The SINAMICS G110s are a range of frequency inverters for controlling the speed of three phase AC motors, incorporating the CPM 110 controlled power module. The various models available range from 120 W to 3.0 kW single-phase input.

The inverters are microprocessor-controlled and use state-of-the-art Insulated Gate Bipolar Transistor (IGBT) technology. This makes them reliable and versatile. A special pulse-width modulation method with selectable pulse frequency permits quiet motor operation. Comprehensive protective functions provide excellent inverter and motor protection.

The SINAMICS G110 CPM110 with its default factory settings is ideal for a large range of simple V/f motor control applications. Using the comprehensive range of programmable parameters provided with the inverter, the unit can be adapted for a wide range of applications. Parameters can be changed using either USS communications or the Basic Operator Panel (BOP).

The SINAMICS G110 is available in two variants; the Analog controlled variant and the USS controlled variant utilizing RS485 protocol. They are available as filtered and unfiltered inverters including a "Flat Plate" version which completes the range. They can be used in both 'stand-alone' applications as well as being integrated into 'Automation Systems".

1.2 Features

Main Characteristics

- Easy installation
- Easy commission
  - Quick commissioning
  - Reset function (allowing the reset of all values to the preset factory defaults)
- Rugged EMC design
- Can be operated on IT line supplies (unfiltered variants)
- 1 digital output – Isolated optocoupler
- 3 digital inputs (non-isolated)
- 1 Analog input, AIN: 0 – 10 V (Analog variant only). Can be used as 4th digital input
- High pulse frequencies for low-noise motor operation
- Status information and alarm messages using the Basic Operator Panel
- Optional Basic Operator Panel with the capability to clone parameter sets
- USS communications interface (USS variant only)
- PC to RS232 Connection Kit available

The inverter can be screened using the methodology shown in Figure 2-8 below.

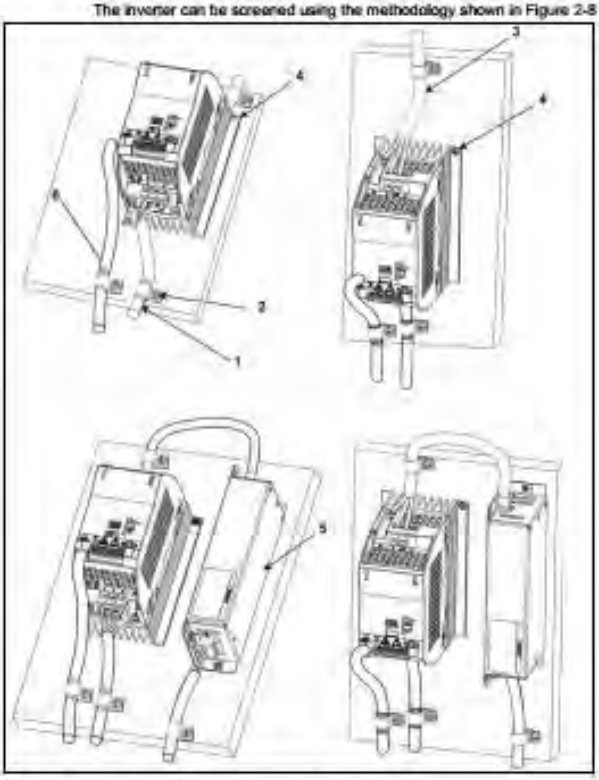


Figure 2-8 Wiring Guidelines to Minimize the Effects of EMI

Legend

1 Motor cable
2 Use suitable clips to fix motor and control cable screens securely to metal back plate
3 Mains power cable
4 Retaining screw (inverter fixing bolts)
5 Line commutating Choke
6 Control cable

2.10 SINAMICS G110 Flat Plate Variant

The SINAMICS G110 Flat Plate variant has been designed to allow greater flexibility in the installation of the inverter by an advanced user as an individual unit or as part of an automation system. Adequate measures must be taken to ensure the correct heat dissipation, which may require an additional external heatsink outside the electrical cabinet enclosure (see Table 2-3 on page 30).

The installation procedures, both mechanical and electrical, starting on page 17 of these Operating Instructions should be performed unless otherwise stated below. Ensuring that all warnings and cautions given throughout the procedures are strictly observed.

Figure 2-9 SINAMICS G110 Flat Plate Variant

WARNING

Operation with the input voltages less than 230 V and 50 Hz or with a pulse frequency greater than 8 kHz will cause an additional heat load on the inverter. These factors must be taken into account when designing the installation conditions and must be verified by a practical load test.

Cooling Considerations

1. For the correct dimensioning of the panel, please consult the panel builder and his technical documentation.
2. For the correct dimensioning of the external heatsink, please refer to the technical data as shown in Table 2-1 on page 20.
3. The rear plate must be able to withstand at least 95°C during fault free operation and carry the heat load (steel or aluminium plate) under the full load conditions and the maximum operating temperatures [-10°C to +50°C (14°F to 122 °F)]. For further information see Table 2-3 on page 30.
4. The minimum clearance distances of 30 mm at both sides and 100 mm vertical spacing at both sides must be observed.
5. It is recommended that the mounding area of the rear plate has to be, as a minimum, at least the same area of the flat plate on the inverter.
6. Side by side or stacked mounting is not allowed for the SINAMICS G110 Flat Plate inverter.

Installation

1. Prepare the mounting surface for the inverter using the dimensions given in Table 2-1 on page 20.
2. Ensure that any rough edges are removed from the drilled holes.
3. Ensure the inverter's Flat Plate is clean and free from dust and grease.
4. The mounting surface for the Flat Plate and if applicable the external heatsink must be:
   - Clean and free from dust and grease.
   - Smooth.

- Free from dents and holes.
- Made of metal (steel or aluminium).
- Must not be painted.
- Must be free from rust.

5. Apply coating of heatsink/thermal contact paste to the inverter's flat plate.
6. Ensure that the contact paste is spread evenly over the rear surface of the Flat Plate.
7. Mount the inverter using four M4 screws.
8. Ensure that the inverter is mounted securely and the M4 screws are lightened to the correct torque of 2.5 Nm (22.12 lbf.in).
9. If required, connect the external heatsink on the other side of the rear plate, ensuring an evenly spread thermal contact paste is apply.
10. When the installation is completed the effectiveness of the cooling should be verified by a load test.
11. Check that there is no trip F0004.

Table 2-3 Flat Plate Power Losses and Thermal Specifications*

	120 W	250 W	370 W	550 W	750 W
Operating temperature range (°C)	-10 to +50				-10 to +40
Total losses (W)	22	28	36	43	54
Linesicle and control losses (W)	9	10	12	13	15
Recommended thermal resistance of heatsink (K/W)	3.0	2.2	1.6	1.2	1.2
Recommended output current (A)	0.9	1.7	2.3	3.2	3.9

*The losses given in Table 2-3 are applicable for units fitted with 25 m of screened cable

3.2 Commission Modes

Basic commissioning of the SINAMICS G110 inverter can be performed by using one of the following methods which are suitable for a large range of applications:

- Using the inverter with its factory default settings by connecting analogue and digital inputs or using the RS485 connections (see Section 3.3.1 on page 35).
- Using the optional Basic Operator Panel (BOP) (see Section 3.3.2. on page 37).

Advanced commissioning allows the user to suitably adapt the inverter to the specific application. Section 3.4 contains information about:

- Using a PLC to communicate with the inverter via the USS protocol (see Section 3.4.1 on page 42).
- Using the PC Tool "STARTER" which communicates with the inverter via USS protocol (see Section 3.4.1 on page 43).
- Optimal configuration of the inverter by setting parameters using 'Quick Commissioning' (see Section 3.4.4 on page 46).
- Resetting inverter parameters to factory default (see Section 3.4.5 on page 49).
- Connecting a PTC temperature sensor to the inverter (see Section 3.4.6 on page 49).
- Parameter Cloning with the BOP (see Section 3.4.7 on page 49).

Notes

When utilizing USS communications, a common 0 V reference connection is required between all devices on the USS bus. Terminal 10 on the control board can be used for this purpose.

The SINAMICS G110 inverter is available as two variants:

1. **Analog Variant**
   The analog variant is suited to stand-alone applications. This variant is designed to be controlled using external switches and a potentiometer utilizing the analog input and digital inputs. Switches and potentiometers are not provided as standard with the inverter.
2. **USS Variant**
   The USS variant is suited to inverter networks. This variant is designed to be controlled utilizing the USS protocol via the RS485 communications interface. A number of inverters can then be connected and controlled via the same communications bus.

The variant can be identified by reading the MLFB from the rating label and comparing it to the MLFBs listed in Table 7-4 and Table 7-5 on pages 68 and 69 respectively.

Since the SINAMICS G110 is available in two variants; different options for the commissioning of the inverters are available to the user. These commissioning options are described in the following sections covering the 'Basic Commissioning' and 'Advanced Commissioning'.

3.3 Basic Commissioning

The SINAMICS G110 is supplied with default parameter settings to cover the following basic operation:

- The motor rating data; voltage, current and frequency data has already been keyed into the inverter to ensure that the motor is compatible with the inverter. (A Siemens standard motor is recommended).
- Linear V/f motor speed, controlled by an analogue potentiometer, or via the RS485 connection using the USS variant.
- Maximum speed 3000 min^{-1} corresponding to a 2-pole motor with 50 Hz (3600 min^{-1} with 60 Hz); controllable using a potentiometer via the inverter's analogue input, or via the RS485 connection using the USS variant.
- Ramp-up time / Ramp-down time = 10 s.

Changing the motor base frequency

The default motor base frequency of the SINAMICS G110 inverter is 50 Hz. In some parts of the world motors are designed for a base frequency of 60 Hz. Changing the motor default base frequency is accomplished using the DIP switch which is provided on the front of the inverter to select the required base frequency.

A small screwdriver will be required to change the position of the DIP switches.

DIP switch 1 is used to change the motor base frequency of the inverter, by default it is in the position '50 Hz'. See Figure 3-2. In the default position (50 Hz) the output power will be displayed in kW (if a BOP is fitted to the inverter). In addition, all motor related parameters will be calculated using the 50 Hz setting.

To change the motor base frequency to 60 Hz, DIP switch 1 must be set to the position '60 Hz'.

The DIP switch must be set to the required frequency before power is applied to the inverter. On power up the inverter will read the DIP switch setting and calculate the following motor related parameters:

- Rated Motor Frequency (P0310)
- Maximum Motor Frequency (P1082)
- Reference Frequency (P2000)

Figure 3-2 Motor Base Frequency DIP Switch and Bus Termination

3.3.1 Factory Settings

The inverter has already been programmed at the factory for standard V/f applications on a Siemens standard four-pole 3-phase induction motor, that has the same power rating as the inverters.

Controlling the speed of the motor is accomplished by connecting the analog inputs on the analog variant (switches and the potentiometer are not supplied with the inverter) or via the RS485 connections on the USS variant as shown in Figure 3-3 below.

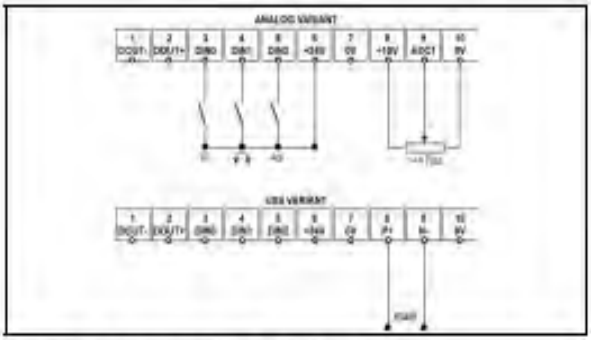


Figure 3-3 Basic operation – Analog and USS Variants

The inverter can be used with its default settings for a wide range of applications. The default settings are shown in Table 3-1 (Analog variant) and Table 3-2 (USS variant). The terminal layout is shown in Figure 3-3 above.

Note

The motor base frequency might have to be changed as described in the previous section on page 34.

Table 3-1 Factory settings for operation using the standard inverter (Analog Variant)

Description	Terminals	Parameter Default	Default Operation
Frequency Setpoint Source	9	P1000 = 2	Analog Input
Command Source	3,4 & 5	P0700 = 2	(see below)
Digital Input 0	3	P0701 = 1	ON/OFF1
Digital Input 1	4	P0702 = 12	Reverse
Digital Input 2	5	P0703 = 9	Fault Acknowledge
Control Method	-	P0727=0	Siemens Standard Control

With the default settings of the inverter (Analog variant) the following is possible:

- Start and stopping the motor (DIN0 via external switch)
- Reversing the motor (DIN1 via external switch)
- Fault Acknowledgement (DIN2 via external switch).

Bus termination on USS variant

The USS variant of the SINAMICS G110 inverter uses RS485 protocols to communicate with the controlling system and all other inverters connected to the network.

It is necessary to terminate the last inverter on the network bus. This is achieved by setting the Bus Termination DIP switches on the front of the inverter to the 'Bus Termination' position (as shown in Figure 3-2 on page 34). It is important that both DIP switches (2 and 3) are set to the 'Bus Termination' position (not in the OFF position). A small screwdriver will be required to change the position of the DIP switches.

3.3.2 Commissioning with the Optional Basic Operator Panel

If the optional 'Basic Operator Panel' (BOP) is available, the control signals and speed reference can easily be set by pressing the relevant buttons. The BOP also provides easy access to the inverter parameters. This section describes how to commission and start the inverter with the minimum of effort, using the BOP.

For advanced use of the BOP to perform for example, the full inverter commissioning see Section 3.4.1 on page 42 or refer to Section 3.4.7 on page 49 for information about parameter cloning using the BOP.

For instructions on how to fit the BOP to the inverter, please refer to Appendix C on page 82 and for a description of the buttons see Appendix D on page 83.

- The BOP must be connected directly to the inverter and not remotely connected via a cable.
- The BOP can be fitted to and removed from the inverter whilst power is applied.
- The inverter will automatically recognize that the BOP has been fitted to the inverter and give the user access to the parameters. To run the inverter (start/stop, setpoint) using the BOP, parameters P0700 (command source, i.e. start/stop, reverse, jog) and P1000 (frequency setpoint) have to be set to 1. Alternatively, P0719 can be set to 11 which is described below.

Figure 3-4 BOP

Note

The motor base frequency might have to be changed as described in the previous section on page 34.

Changing parameters with the Basic Operator Panel

The description below serves as an example that shows how to change any parameters using the BOP. These examples can also be used as a guide to configuring the inverter for running via the BOP (start/stop commands and the frequency setpoint are input on the BOP).

Changing P0003 – parameter access level

	Step	Result on display
1	Press P to access parameters	r0000
2	Press ▲ until P0003 is displayed	P0003
3	Press P to display the parameter value	1
4	Press ▲ or ▼ to set the required value (set to 3)	3
5	Press P to confirm and store the value	P0003
6	All level 1 to level 3 parameters are now visible to the user.	

Changing P0719 an indexed parameter – setting BOP control

	Step	Result on display
1	Press P to access parameters	r0000
2	Press ▲ until P0719 is displayed	P0719
3	Press P to access the parameter value	in000
4	Press ▲ or ▼ to select index 1	in001
5	Press P to display actual set value	0
6	Press ▲ or ▼ to the required value	11
7	Press P to confirm and store the value	P0719
8	Press ▼ until r0000 is displayed	r0000
9	Press P to return the display to the standard drive display (as defined by the customer)	

Figure 3-5 Changing parameters via the BOP

NOTE

In some cases - when changing parameter values - the display on the BOP shows BUSY. This means the inverter is busy with tasks of a higher priority.

Changing single digits in parameter values

For changing the parameter value rapidly, the single digits of the display can be changed by performing the following actions:

1. Ensure you are in the parameter value changing level (see section "Changing parameters with Basic Operator Panel" above).
2. Press Fn (function button), which causes the right hand digit to blink.
3. Change the value of this digit by pressing ▲ or ▼.
4. Press Fn (function button) again causes the next digit to blink.
5. Perform steps 2 to 4 until the required value is displayed.
6. Press the P to leave the parameter value changing level.

NOTE

The function button Fn may also be used to acknowledge a fault condition.

Commissioning of motorpoti (MOP) function

Simple motor speed control can be achieved using the motorpoti (MOP) function of the optional BOP (for using the MOP see also P1031 and P1040 in the Parameter List).

The BOP motor control functions are disabled by default. To control the motor via the BOP, the following settings must be completed (see also "Changing parameters with the Basic Operator Panel" above):

➢ P0719 = 11 (enables start/stop button on the BOP and enables the motor potentiometer setpoint from the BOP).

Alternatively set:

- P0700 = 1 (enables the start/stop button on the BOP).
- P1000 = 1 (this enables the motor potentiometer setpoints).

1. Press the I button to start the motor.
2. Press the ▲ button while the motor is turning. Motor speed increases to 50 Hz.
3. When the inverter reaches 50 Hz, press button ▼. Motor speed and displayed value are decreased.
4. Change the direction of rotation by pressing the ↔ button.
5. Stop the motor by pressing the O button.

If the BOP has been set as the command source (P0700 = 1 or P0719 = 10 - 15), the inverter will stop if the BOP is removed.

3.4.4 Quick Commissioning (P0010=1)

Quick commissioning is an easy way to optimally configure the SINAMICS G110 inverter to a specific motor. The motor data, taken from the motor rating label, is entered into the inverter and then the inverter calculates the dependent control and protection parameters.

An alternative to quick commissioning is parameter cloning (see page 49) that can be used if a large number of inverters are to be commissioned to the same specific motor.

NOTE

It is only possible to change motor parameters when quick commissioning is enabled (P0010=1).

It is important that parameter P0010 is used for commissioning and P0003 is used to select the number of parameters to be accessed. The P0003 parameter allows a group of parameters to be selected that will enable quick commissioning. Parameters such as Motor settings and Ramp settings are included.

At the end of the quick commissioning sequence, P3900 should be selected, which, when set to 1, will carry out the necessary motor calculations and clear all other parameters (not included in P0010=1) to the default settings. This will only happen in the Quick Commissioning mode.

Parameter P0010 is automatically reset to the value 0 when P3900 > 0. The inverter can only be run if P0010 has been set back to 0.

NOTE

We recommend commissioning is performed according to this scheme. Nevertheless an expert user is allowed to perform the commissioning with the filter functions of P0004.

Motor data for parameterization

Figure 3-6 below indicates where to find the relevant motor data on the motor rating plate. Figure 3-6 is for illustration purposes only and the values within this figure should not be keyed into your inverter, rather the values from your own motor's rating plate should be keyed into the inverter.

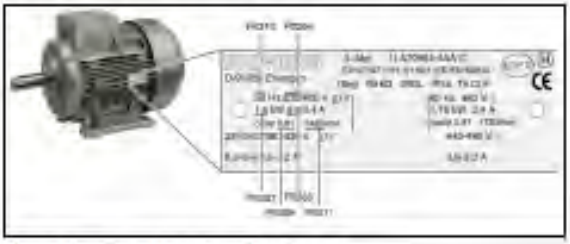
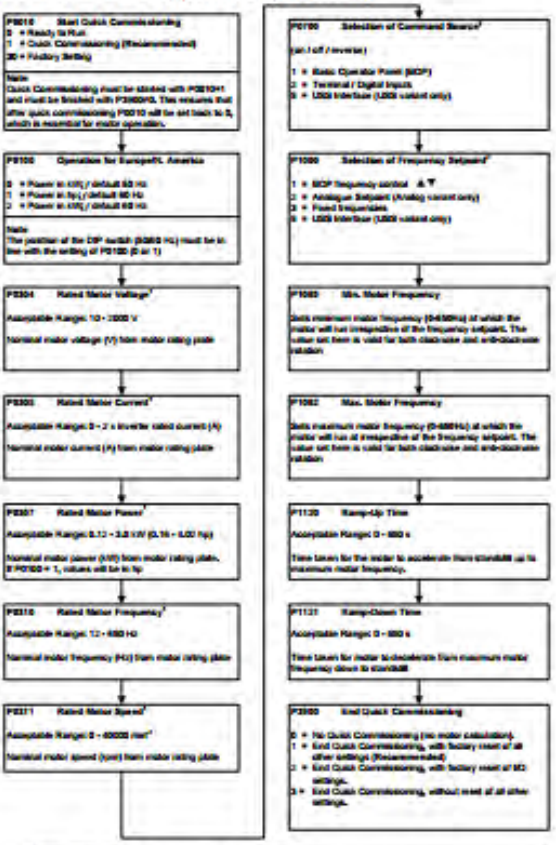


Figure 3-6 Typical Motor Rating Plate Example

Flow chart Quick Commissioning (Level 1 Only – P0003=1)



1. Motor related parameters - please refer to the motor rating plate.
2. Indicates parameters that contain more detailed lists of possible settings for use in specific applications. Please refer to the Parameter List.

D Description of the BOP

Panel/Button	Function	Effects
r0000	Indicates Status	The LCD displays the settings currently used by the converter.
I	Start motor	Pressing the button starts the inverter. This button is disabled by default. To enable this button set P0700 = 1.
O	Stop motor	OFF1 Pressing the button causes the inverter to come to a standstill at the selected ramp down rate. OFF2 Pressing the button twice (or once long) causes the motor to coast to a standstill. This function is always enabled.
	Change direction	Press this button to change the direction of rotation of the motor. Reverse is indicated by a minus (-) sign or a flashing decimal point. Disabled by default, to enable set P0700 = 1.
JOG	Jog motor	Pressing this button while the inverter has no ON command causes the motor to start and run at the preset jog frequency. The inverter stops when the button is released. Pressing this button when the inverter/motor is running has no effect.
FN	Functions	This button can be used to view additional information. Pressing and holding the button for 2 seconds from any parameter during operation, shows the following: 1. DC link voltage (indicated by d – units V). 2. Output voltage (indicated by o – units V). 3. Output frequency (Hz). 4. The value selected in P0005. Additional presses will toggle around the above displays. A short press of the button will acknowledge a fault condition. **Jump Function** From any parameter (rXXXX or PXXXX) a short press of the Fn button will immediately jump to r0000, you can then change another parameter, if required. Upon returning to r0000, pressing the Fn button will return you to your starting point.
P	Access parameters	Pressing this button allows access to the parameters.
▲	Increase value	Pressing this button increases the displayed value.
▼	Decrease value	Pressing this button decreases the displayed value.

2. *Datasheet* Ethernet Board WIZ110SR

1.2 Specification

Items	Description
MCU	8051 Compliant (having internal 62K Flash, 16K SRAM, 2K EEPROM)
TCP/IP	W5100 (Ethernet PHY Embedded)
Network Interface	10/100 Mbps auto-sensing RJ-45 Connector
Serial Interface	RS232
Serial Signal	TXD, RXD, RTS, CTS, GND
Serial Parameters	Parity : None, Even, Odd
	Data Bits : 7,8
	Flow Control : None, RTS/CTS, XON/XOFF
	Speed : up to 230Kbps
Input Voltage	DC 5V
Power Consumption	Under 180mA
Temperature	0°C ~ 80°C (Operation), -40°C ~ 85°C (Storage)
Humidity	10 ~ 90%

Table 1. WIZ110SR Specification

1.3 Products Contents

	WIZ110SR Board
	CD (Option / including Firmware, Configuration Tool Program Manual and other related materials)

5. Hardware Description of WIZ110SR

5.1 Parameters

- Power 5V DC / 180mA
- Dimension 75 x 50 x 17 (L x W x H)
- Temperature Operating : 0 ~ 80 ℃
- Ethernet 10/100 Base-T Ethernet (Auto detection)
- Serial interface RS-232

5.2 Specification

- MCU 8051 Compliant
- FLASH 62KByte (MCU Internal)
- SRAM 16KByte (MCU Internal)
- EEPROM 2KByte (MCU Internal)

5.3 Board Dimensions and Pin Assignment

5.3.1 Dimensions

2. WIZ110SR Board

2.1 Block Diagram

Figure 1. Block Diagram

WIZ110SR is a protocol converter that transmits the data sent by serial equipment as TCP/IP data type and converts back the TCP/IP data received through the network into serial data to transmit back to the equipment. When the data is received from serial port, it is sent to W5100 by MCU. If any data is transmitted from Ethernet, it is received in the internal buffer if W5100, and sent to the serial port by MCU. MCU in the module controls the data according to the configuration value that user defined.

<TCP server mode>

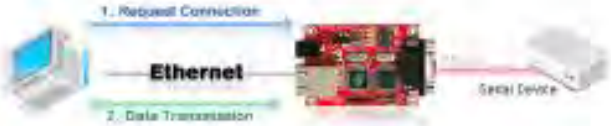


Figure 3. TCP Server mode

At the TCP Server mode, WIZ110SR waits for the connection requests.
TCP Server mode can be useful when the monitoring center tries to connect to the device (where WIZ110SR is installed) in order to check the status or provide the commands. In normal time WIZ110SR is on the waiting status, and if there is any connection request from the monitoring center, data communication is processed and connection is closed.

In order to operate this mode, Local IP, Subnet, Gateway Address and Local Port Number should be configured first.

As illustrated in the above figure, data transmission proceeds as follows,
1. The host connects to the WIZ1q0SR which is configured as TCP Server mode.
2. As the connection is established, data can be transmitted in both directions - from the host to the WIZ110SR, and from the WIZ110SR to the host

<TCP client mode>

Figure 4. TCP Client mode

If WIZ110SR is set as TCP Client, it tries to establish connection to the server.
To operate this mode, Local IP, Subnet, Gateway Address, Server IP, and Server port number should be set. If server IP had domain name, use DNS function.

In TCP Client mode, WIZ110SR can actively establish a TCP connection to a host computer when power is supplied.

As illustrated in the above figure, data transmission proceeds as follows:
1. As power is supplied, WIZ110SR board operating as TCP client mode actively establishes a connection to the server.
2. If the connection is complete, data can be transmitted in both directions - from the host to the WIZ110SR and from WIZ110SR to the host

<Mixed mode>
In this mode, WIZ110SR normally operates as TCP Server and waits for the connection request from the peer. However, if WIZ110SR receives data from the serial device before connection is established, it changes to the client mode and sends the data to the server IP. Therefore, at the mixed mode, the server mode is operated prior to the client mode.

As like TCP Server mode, the Mixed mode is useful for the case that the monitoring center tries to connect to the serial device (in which WIZ110SR is used) to check device status. In addition to this, if any emergency occurs in the serial device, the module will change to Client mode to establish the connection to the server and deliver the emergency status of the device

❺ Use UDP mode

Figure 5. UDP mode

At the UDP mode, the connection establishment is not defined. Just set the IP address and port number of the peer and send the data.

4. Demonstration and Test

This chapter will show how WIZ110SR operates through demonstration and test. The hardware and software for the testing is shown in below table.

	PC	WIZ110SR
Hardware	1) RS232 Port 2) LAN Port	1) WIZ110SR Board 2) Serial Cable 3) LAN Cable 4) DC5V Power Adaptor
Software	1) Configuration Tool Program 2) Hyper Terminal	

4.1 Hardware Interface

Figure 12. WIZ110SR Interface

Install the board as below steps.

STEP 1: Connect the WIZ110SR to the network by using RJ45 Ethernet cable.

STEP 2: Connect the WIZ110SR to the serial device by using Serial cable

STEP 3: Connect the power adaptor to WIZ110SR board.

3. ***Datasheet* Sensor *Rotary Encoder***

E50S Series

Diameter ø50mm Shaft type Incremental Rotary encoder

Line-up

Features

- 12-24VDC power supply of line driver output(Line-up)
- Suitable for measuring angle, position, revolution, speed, acceleration and distance
- Power supply : 5VDC, 12-24VDC ±5%

Applications

- Various tooling machinery, packing machine and general industrial machinery etc.

Please read "Caution for your safety" in operation manual before using.

Ordering information (Former name : ENB)

E50S - 8 - 5000 - 3 - N - 24 - [ ]

Series	Shaft diameter	Pulse/1Revolution	Output phase	Output	Power supply	Cable
Diameter ø50mm, shaft type	ø8mm	Refer to resolution	2: A, B 3: A, B, Z 4: A, $\overline{A}$, B, $\overline{B}$ 6: A, $\overline{A}$, B, $\overline{B}$, Z, $\overline{Z}$	T: Totem pole output N: NPN open collector output V: Voltage output L: Line driver output	5: 5VDC ±5% 24: 12-24VDC ±5%	No mark: Without type T: Cable outgoing connector type(-) C: Cable side outgoing connector integrated type CS: Cable outgoing connector integrated type

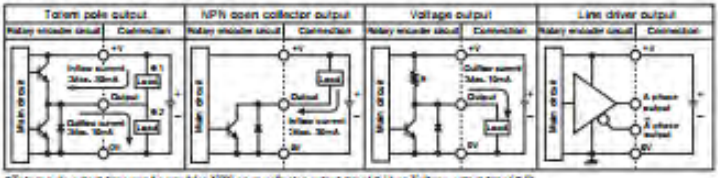
- Standard : E50S8-5000-3-N-24
- Cable length : 2m

Specifications

Item			Diameter ø50mm shaft type of incremental rotary encoder
Resolution(P/R)			(Note1) 1, 2, 5, 10, 12, 15, 20, 30, 36, 50, 60, 75, 100, 120, 125, 150, 180, 200, 240, 250, 254, 300, 360, 400, 500, 600, 720, 800, 900, 1000, 1024, 1200, 1250, 1440, 2000, 2048, 2500, 3000, 3600, 4000, 5000, 6000
Electrical specification	Output phase		A, B, Z phase(Line driver : A, $\overline{A}$, B, $\overline{B}$, Z, $\overline{Z}$ phase)
	Phase difference of output		Phase difference between A and B : $\frac{T}{4} \pm \frac{T}{8}$ (T=1cycle of A phase)
	Control output	Totem pole output	• Low : Load current:Max. 30mA, Residual voltage : Max. 0.4VDC • High : Load current:Max. 10mA, Output voltage(Power supply 5VDC):Min. (Power supply-2.0)VDC, Output voltage(Power supply 12-24VDC):Min. (Power supply-3.0)VDC
		NPN open collector output	Load current : Max. 30mA, Residual voltage : Max. 0.4VDC
		Voltage output	Load current : Max. 10mA, Residual voltage : Max. 0.4VDC
		Line driver output	• Low : Load current : Max. 20mA, Residual : Max. 0.5VDC • High : Load current : Max. -20mA, Output voltage(Power supply 5VDC) : Min. 2.5VDC, Output voltage(Power voltage 12-24VDC) : Min. (Power supply-3.0)VDC
	Response time (Rise/Fall)	Totem pole output / NPN open collector output / Voltage output / Line driver output	Max. 1μs(Cable length : 2m, (tank = 20mA))
	Max. response frequency		300kHz
	Power supply		• 5VDC ±5%(Ripple P-P : Max. 5%) • 12-24VDC ±5%(Ripple P-P : Max. 5%)
	Current consumption		Max. 80mA (disconnection of the load), Line driver output : Max. 50mA (disconnection of the load)
	Insulation resistance		Min. 100MΩ (at 500VDC megger between all terminals and case)
	Dielectric strength		750VAC 50/60Hz for 1 minute (between all terminals and case)
	Connection		Cable outgoing type, 2000mm cable outgoing connector type, Connector integrated type(Note6, Note7)
Mechanical specification	Starting torque		Max. 10gf · cm(0.007N · m)(Note2) / Max. 300gf · cm(0.029N · m)(Note3)
	Moment of inertia		Max. 30g · cm^2(5 × 10^{-6}kg · m^2)(Note2) / Max. 300g · cm^2(5.1 × 10^{-5}kg · m^2)(Note3)
	Shaft loading		Radial : 10kgf, Thrust : 2.5kgf
	Max. allowable revolution	(Note4)	5000rpm
Vibration			1.5mm amplitude at frequency of 10 to 55Hz(for 1 min.) in each of X, Y, Z directions for 2 hours
Shock			Max. 75G
Ambient temperature			-10 to 70°C(at non-freezing status), Storage : -25 to 85°C
Ambient humidity			35 to 85%RH, Storage : 35 to 90%RH
Protection			Normal type, Cable outgoing connector type : IP50(IEC standard)(Note5), Connector integrated type : IP40(IEC standard)
Cable			ø5mm, 5P, Length : 2m, Shield cable(Line driver output : ø6mm, 8P) (AWG 20, Core wire diameter : 0.09mm, No. of core wire : 20, Insulator out diameter : ø1.1mm)
Accessory			ø8mm coupling, bracket
Approval			Normal type : CE (Except for line driver output)
Unit weight			Approx. 270g, Connector integrated type : 190g

- (Note1) "-" pulse is only for A, B phase(Line drive output is for A, $\overline{A}$, B, $\overline{B}$ phase).
- (Note2) This value is for normal type, cable outgoing connector type (Protection: IP50).
- (Note3) This value is for normal type, cable outgoing connector type (Protection: IP50)(connector integrated type (Protection: IP40)).
- (Note4) Make sure that max. response revolution should be lower than or equal to max. allowable revolution when selecting the resolution.
  (Max. response revolution(rpm)) = $\frac{\text{Max. response frequency}}{\text{Resolution}} \times 60\text{ sec.}$
- (Note5) "Normal type, cable outgoing connector type is quotet as IP54 protection.

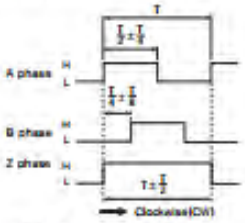
■Control output diagram



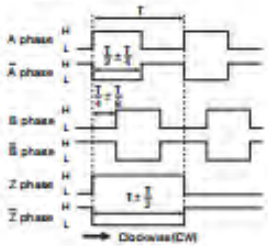
※Totem pole output type can be used for NPN open collector output type(※1) or Voltage output type(※2).
※The output circuit of A, B, Z phase are the same, Line driver output is A, Ā, B, B̄, Z, Z̄.

■Output waveform

※Totem pole output / NPN open collector output / Voltage output



※Line driver output

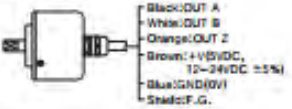


※CW : Right turn as from the shaft

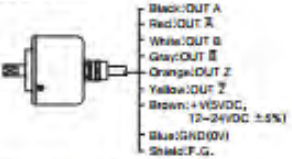
■Connections

■Normal type

※Totem pole output / NPN open collector output / Voltage output



※Line driver output



※Unused wires must be insulated.
※The shield cable and metal case of encoder must be grounded(F.G.)

■Cable outgoing connector/ Connector integrated type

※Totem pole output / NPN open collector output / Voltage output

※Line driver output

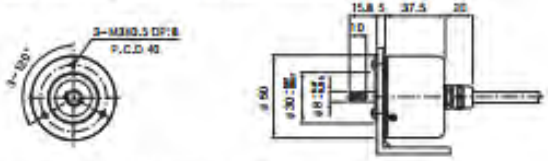
Totem pole output / NPN open collector output / Voltage output			Line driver output		
Pin No	Function	Cable color	Pin No	Function	Cable color
①	OUT A	Black	①	OUT A	Black
②	OUT B	White	②	OUT Ā	Red
③	OUT Z	Orange	③	+V	Brown
④	+V	Brown	④	GND	Blue
⑤	GND	Blue	⑤	OUT B	White
⑥	F.G.	Shield	⑥	OUT B̄	Gray
			⑦	OUT Z	Orange
			⑧	OUT Z̄	Yellow
			⑨	F.G.	Shield

※F.G.(Field Ground) : It must be grounded separately.

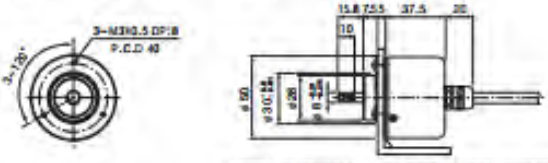
●Dimensions

(Unit:mm)

■Normal type, Cable outgoing connector type(Protection : IP50)



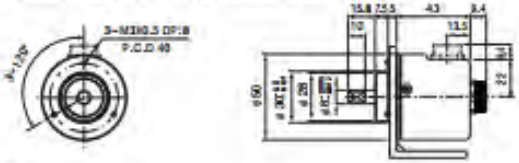
■Normal type, Cable outgoing connector type(Protection : IP64)



Cable for normal type	Cable for Cable outgoing connector type
ø 5mm, SP:Line driver output(5P), Length:3000mm, Shield cable	ø 5mm, SP:Line driver output(5P), Length:350mm, Shield cable

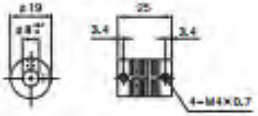
※ Connector cable is sold separately and see C-6 for specifications.

■Rear/Side connector integrated type(Protection: IP65)



●Bracket

●Coupling(E50S)



- Parallel misalignment: Max. 0.2mm
- Angular misalignment: Max. 3°
- End-play: Max. 0.2mm

※For parallel misalignment, angular misalignment, End-play terms, refer to F-69 page.

※For Encoder coupling(E50) detail information, refer to F-68 page.

※ may result in damage to this unit.

■ Outline

This unit is very useful to control length, angle and position by converting revolution value of shaft into number of pulse as an optical incremental Encoder.

■ Ordering information

ENA	- 5000	- 2	- N	- 24
Series	Pulse/1Revolution	Output phase	Output	Power supply
Shaft type to be mounted at the side (Shaft diameter ø10mm)	See resolution	2: A, B 3: A, B, Z	T: Totem Pole output N: NPN open collector output V: Voltage output	5: 5VDC±5% 24: 12~24VDC ±5%

※Standard: ENA-PULSE-2-N-24 ※Standard: A, B

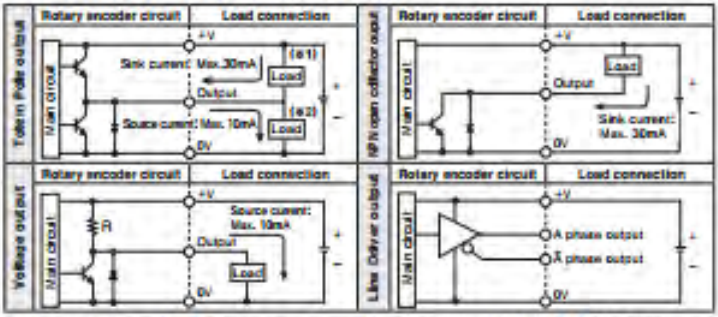
E50S	8	- 8000	- 3	- N	- 24	-
Series	Shaft diameter	Pulse/1Revolution	Output phase	Output	Power supply	Cable
Diameter ϕ50mm shaft type	ø8mm	See resolution	2: A, B 3: A, B, Z 4: A, $\overline{A}$, B, $\overline{B}$ 6: A, $\overline{A}$, B, $\overline{B}$, Z, $\overline{Z}$	T: Totem Pole output N: NPN open collect output V: Voltage output L: Line Driver ouput	5: 5VDC±5% 24: 12~24VDC ±5%	No mark: Normal type (※)C: Cable outgoing connector type

※Standard: E50S8-PULSE-3-N-24 ※Cable length:250mm

ENC	- 1	- 1	- N	- 24	-
Series	Output phase	Min. measuring unit	Output	Power supply	Cable
Wheel type	1: A, B	1: 1mm 2: 1cm 3: 1m 4: 0.01yd 5: 0.1yd 6: 1yd	T: Totem Pole output N: NPN open collector output V: Voltage output	5: 5VDC±5% 24: 12~24VDC ±5%	No mark: Normal type (※)C: Cable outgoing connector type

※Cable length:250mm

■ Control output diagram



※ The output circuit of A, B, Z phase are the same.(Line Driver output is A, $\overline{A}$, B, $\overline{B}$, Z, $\overline{Z}$ phase)
※ Totem Pole output can be used for NPN open collector type(※1) or voltage output type(※2).

※ The above specification are subject to change without notice.

4. ***Datasheet* IC MAX232**

TEXAS INSTRUMENTS

MAX232, MAX232I

SLLS047M – FEBRUARY 1989 – REVISED NOVEMBER 2014

MAX232x Dual EIA-232 Drivers/Receivers

1 Features

- Meets or Exceeds TIA/EIA-232-F and ITU Recommendation V.28
- Operates From a Single 5-V Power Supply With 1.0-μF Charge-Pump Capacitors
- Operates up to 120 kbit/s
- Two Drivers and Two Receivers
- ±30-V Input Levels
- Low Supply Current: 8 mA Typical
- ESD Protection Exceeds JESD 22
  - 2000-V Human-Body Model (A114-A)
- Upgrade With Improved ESD (15-kV HBM) and 0.1-μF Charge-Pump Capacitors is Available With the MAX202 Device

2 Applications

- TIA/EIA-232-F
- Battery-Powered Systems
- Terminals
- Modems
- Computers

3 Description

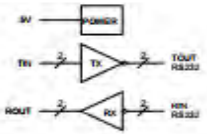
The MAX232 device is a dual driver/receiver that includes a capacitive voltage generator to supply TIA/EIA-232-F voltage levels from a single 5-V supply. Each receiver converts TIA/EIA-232-F inputs to 5-V TTL/CMOS levels. These receivers have a typical threshold of 1.3 V, a typical hysteresis of 0.5 V, and can accept ±30-V inputs. Each driver converts TTL/CMOS input levels into TIA/EIA-232-F levels.

Device Information[1]

ORDER NUMBER	PACKAGE (PIN)	BODY SIZE
MAX232x	SOIC (16)	9.90 mm × 3.91 mm
	SOIC (16)	10.30 mm × 7.50 mm
	PDIP (16)	19.30 mm × 6.35 mm
	SOP (16)	10.3 mm × 5.30 mm

(1) For all available packages, see the orderable addendum at the end of the datasheet.

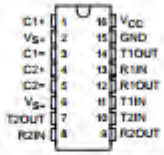
4 Simplified Schematic



6 Pin Configuration and Functions

Top View

MAX232 . . . D, DW, N, OR NS PACKAGE
MAX232I . . . D, DW, OR N PACKAGE
(TOP VIEW)



Pin Functions

PIN NAME	PIN NO.	TYPE	DESCRIPTION
C1+	1	—	Positive lead of C1 capacitor
VS+	2	O	Positive charge pump output for storage capacitor only
C1−	3	—	Negative lead of C1 capacitor
C2+	4	—	Positive lead of C2 capacitor
C2−	5	—	Negative lead of C2 capacitor
VS−	6	O	Negative charge pump output for storage capacitor only
T2OUT, T1OUT	7, 14	O	RS232 line data output (to remote RS232 system)
R2IN, R1IN	8, 13	I	RS232 line data input (from remote RS232 system)
R2OUT, R1OUT	9, 12	O	Logic data output (to UART)
T2IN, T1IN	10, 11	I	Logic data input (from UART)
GND	15	—	Ground
VCC	16	—	Supply Voltage, Connect to external 5V power supply

7 Specifications

7.1 Absolute Maximum Ratings(1)

over operating free-air temperature range (unless otherwise noted)

			MIN	MAX	UNIT
V_{CC}	Input Supply voltage range(2)		−0.3	6	V
V_+	Positive output supply voltage range		V_{CC} − 0.3	15	V
V_-	Negative output supply voltage range		−0.3	−15	V
V_I	Input voltage range	T1IN, T2IN	−0.3	V_{CC} + 0.3	V
		R1IN, R2IN		±30	
V_O	Output voltage range	T1OUT, T2OUT	V_- − 0.3	V_+ + 0.3	V
		R1OUT, R2OUT	−0.3	V_{CC} + 0.3	
	Short-circuit duration	T1OUT, T2OUT	Unlimited		
T_J	Operating virtual junction temperature			150	°C

(1) Stresses beyond those listed under *Absolute Maximum Ratings* may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated under *Recommended Operating Conditions* is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.
(2) All voltages are with respect to network GND.

7.2 Handling Ratings

			MIN	MAX	UNIT
T_{stg}	Storage temperature range		−65	150	°C
$V_{(ESD)}$	Electrostatic discharge	Human body model (HBM), per ANSI/ESDA/JEDEC JS-001, all pins(1)	0	2000	V
		Charged device model (CDM), per JEDEC specification JESD22-C101, all pins(2)	0	1000	

(1) JEDEC document JEP155 states that 500-V HBM allows safe manufacturing with a standard ESD control process.
(2) JEDEC document JEP157 states that 250-V CDM allows safe manufacturing with a standard ESD control process.

7.3 Recommended Operating Conditions

			MIN	NOM	MAX	UNIT
V_{CC}	Supply voltage		4.5	5	5.5	V
V_{IH}	High-level input voltage (T1IN,T2IN)		2			V
V_{IL}	Low-level input voltage (T1IN, T2IN)				0.8	V
R1IN, R2IN	Receiver input voltage				±30	V
T_A	Operating free-air temperature	MAX232	0		70	°C
		MAX232I	−40		85	

7.4 Thermal Information

THERMAL METRIC(1)		MAX232xD	MAX232xDW	MAX232xN	MAX232xNS	UNIT
		SOIC	SOIC wide	PDIP	SOP	
		16 PINS	16 PINS	16 PINS	16 PINS	
$R_{\theta JA}$	Junction-to-ambient thermal resistance	73	57	67	64	°C/W

(1) For more information about traditional and new thermal metrics, see the *IC Package Thermal Metrics* application report (SPRA953).

7.5 Electrical Characteristics — Device

over recommended ranges of supply voltage and operating free-air temperature (unless otherwise noted) (see Figure 6)

PARAMETER	TEST CONDITIONS(1)	MIN	TYP(2)	MAX	UNIT
I_{CC}	Supply current	V_{CC} = 5.5V, all outputs open, T_A = 25°C		8	10	mA

(1) Test conditions are C1–C4 = 1 µF at V_{CC} = 5 V ± 0.5 V.
(2) All typical values are at V_{CC} = 5 V, and T_A = 25°C.

7.6 Electrical Characteristics — Driver

over recommended ranges of supply voltage and operating free-air temperature (unless otherwise noted)

PARAMETER			TEST CONDITIONS[1]	MIN	TYP[2]	MAX	UNIT
V_{OH}	High-level output voltage	T1OUT, T2OUT	R_L = 3 kΩ to GND	5	7		V
V_{OL}	Low-level output voltage[3]	T1OUT, T2OUT	R_L = 3 kΩ to GND		−7	−5	V
r_o	Output resistance	T1OUT, T2OUT	V_{S+} = V_{S-} = 0, V_O = ±2 V	300			Ω
I_{OS}[4]	Short-circuit output current	T1OUT, T2OUT	V_{CC} = 5.5 V, V_O = 0 V		±10		mA
I_{IS}	Short-circuit input current	T1IN, T2IN	V_I = 0			200	µA

(1) Test conditions are C1–C4 = 1 µF at V_{CC} = 5 V ± 0.5 V.
(2) All typical values are at V_{CC} = 5 V, T_A = 25°C.
(3) The algebraic convention, in which the least-positive (most negative) value is designated minimum, is used in this data sheet for logic voltage levels only.
(4) Not more than one output should be shorted at a time.

7.7 Electrical Characteristics — Receiver

over recommended ranges of supply voltage and operating free-air temperature (unless otherwise noted)

PARAMETER			TEST CONDITIONS[1]	MIN	TYP[2]	MAX	UNIT
V_{OH}	High-level output voltage	R1OUT, R2OUT	I_{OH} = −1 mA	3.5			V
V_{OL}	Low-level output voltage[3]	R1OUT, R2OUT	I_{OL} = 3.2 mA			0.4	V
V_{IT+}	Receiver positive-going input threshold voltage	R1IN, R2IN	V_{CC} = 5 V, T_A = 25°C		1.7	2.4	V
V_{IT-}	Receiver negative-going input threshold voltage	R1IN, R2IN	V_{CC} = 5 V, T_A = 25°C	0.8	1.2		V
V_{hys}	Input hysteresis voltage	R1IN, R2IN	V_{CC} = 5 V	0.2	0.5	1	V
r_i	Receiver input resistance	R1IN, R2IN	V_{CC} = 5 V, T_A = 25°C	3	5	7	kΩ

(1) Test conditions are C1–C4 = 1 µF at V_{CC} = 5 V ± 0.5 V.
(2) All typical values are at V_{CC} = 5 V, T_A = 25°C.
(3) The algebraic convention, in which the least-positive (most negative) value is designated minimum, is used in this data sheet for logic voltage levels only.

7.8 Switching Characteristics

over recommended ranges of supply voltage and operating free-air temperature (unless otherwise noted)

PARAMETER		TEST CONDITIONS[1]	MIN	TYP[2]	MAX	UNIT
SR	Driver slew rate	R_L = 3 kΩ to 7 kΩ, see Figure 4			30	V/µs
SR(t)	Driver transition region slew rate	see Figure 5		3		V/µs
	Data rate	One TOUT switching		120		kbit/s
$t_{PLH(R)}$	Receiver propagation delay time, low- to high-level output	TTL load, see Figure 3		500		ns
$t_{PHL(R)}$	Receiver propagation delay time, high- to low-level output	TTL load, see Figure 3		500		ns

(1) Test conditions are C1–C4 = 1 µF at V_{CC} = 5 V ± 0.5 V.

LAMPIRAN D
INISIALISASI INVERTER

Pengaturan Parameter *Quick Comissioning Inverter Sinamics G110*

No.	*Parameter List*	Pengaturan Parameter
1	P0010 (*Start Quick Commissioning*)	P0010 merupakan parameter pengaturan awal untuk memulai mengubah parameter dari *inverter*. Dengan cara menekan tombol "P" yang ada di *inverter*, maka untuk memulai *Quick Comissioning* dengan memilih "1", untuk menyimpan parameter yang sudah diatur dengan cara menekan tombol "P" pada *inverter*.
2	P003 (*User Access Level*)	P003 merupakan parameter untuk menyetujui penggunaan *Quick Comissioning* jadi parameter ini merupakan parameter pertama yang muncul dalam proses *Quick Comissioning*. Untuk mengakses parameter maka dengan menekan tombol "P", kemudian akan muncul 3 pilihan yang terdapat dalam pengaturan ini. Memilih "1", kemudian menekan kembali tombol "P" untuk menyimpan nilai parameter.
3	P0100 (*Operation for Europe / America*)	P0100 merupakan parameter untuk memilih frekuensi operasi yang akan digunakan untuk menggerakkan motor. Ada 3 pilihan dalam pengaturan ini: *0= Power in kW; f default 50 Hz* *1= Power in hp; f default 60 Hz* *2 = Power in kW; f default 60 Hz* Karena frekuensi yang ada di Indonesia sendiri hanya menyediakan jaringan listrik dengan frekuensi 50Hz sama seperti pada negara-negara Eropa, sehingga diharuskan memilih mode *Europe* yaitu "0"

No.	*Parameter List*	Pengaturan Parameter
4	P0304 (*Rated Motor Voltage*)	P0304 merupakan parameter untuk menentukan nilai suplai tegangan motor, pada bagian ini kisaran yang diperbolehkan yaitu 10 –2000 V. Dalam mengisi parameter ini, nilai yang dimasukkan harus sesuai dengan informasi yang ada pada *name plate* motor. Besar nominal tegangan motor (*Volt*) yang tertera pada *name plate* adalah 380V, sehingga nilai yang harus diisi adalah 380
5	P0305 (*Rated Motor Current*)	P0305 merupakan parameter untuk nilai arus nominal dari motor. Pada bagian ini kisaran yang diperbolehkan yaitu 0 – 2x. Nilai Arus nominal yang ada pada *name plate* motor yaitu 0,62 A, sehingga nilai yang harus diisi adalah 0,62
6	P0307 (*Rated Motor Power*)	P0307 merupakan parameter yang menentukan nilai daya motor. Pada bagian ini kisaran yang diperbolehkan adalah 0,12 – 3,0 *KW* (0,16– 4,02 *HP*). Karenna pada Parameter P0100 kita mengisikan 1 maka nominal daya diisikan dalam bentuk HP, besar nominal daya motor yang kita gunakan yaitu 0,25 HP, sehingga nilai yang harus diisi adalah 0,25
7	P0310 (*Rated Motor Frequency*)	P0310 merupakan parameter yang menentukan nilai frekuensi motor. Pada bagian ini kisaran yang diperbolehkan 12 – 650 Hz. Besar nominal frekuensi motor yang tertera pada *name plate* yaitu 50 Hz.
8	P0311 (*Rated Motor Speed*)	P0311 merupakan pengaturan untuk menentukan nilai kecepatan motor, pada bagian ini kisaran yang diperbolehkan 0 –40000 rpm. Besar nominal kecepatan motor (*rpm*) pada *name plate* yaitu 1310 *rpm*.

No.	*Parameter List*	Pengaturan Parameter
9	P0700 (*Selection of Command Source*)	P0700 merupakan parameter untuk pemilihan sumber perintah, dimana nantinya akan muncul tiga pilihan : *1 = Basic Operator Panel (BOP)* *2 = Terminal / Digital Inputs* *5 = USS Interface (USS variant only)* karena semua pengaturan berasal dari *inverter* itu sendiri tanpa memerlukan perangkat lain, maka memilih angka "1"*Basic Operator Panel*.
10	P1000 (*Selection of Frequency Setpoint*)	P1000 merupakan parameter untuk menentukan pengontrolan frekuensi *inverter*. Ada 4 pilihan ketika kita akan menentukan metode pengontrolan frekuensi pada *inverter* yaitu : *1 = BOP frequency control* *2 = Analogue Setpoint (Analog variant only)* *3 = Fixed frequencies* *5 = USS Interface (USS variant only)* Karena untuk mengendalikan motor tiga fasa menggunakan Mikrokontroler *ATMega16* dimana Mikrokontroler ini memberikan tegangan kerja 0–10 V, maka untuk pengendalian frekuensinya menggunakan pilihan "2", yaitu *analog set point*.
11	P1080 (*Minimum Frequency*)	P1080 merupakan parameter untuk menentukan nilai minimal frekuensi motor dengan kisaran frekuensi 0–650 Hz. Motor yang digunakan diatur minimal frekuensi motor sebesar 0 Hz.
12	P1082 (*Maximum Frequency*)	P1082 merupakan parameter untuk menentukan nilai maksimum frekuensi motor dengan kisaran sebesar 0–650 Hz, dimana motor yang digunakan frekuensinya diatur maksimal sebesar 50 Hz.
13	P1120 (*Ramp-up Time*)	P1120 merupakan parameter untuk menentukan nilai *ramp-up time. Ramp-up time* adalah waktu yang dibutuhkan oleh motor dari keadaan diam sampai frekuensi motor maksimum. Waktu yang dibutuhkan untuk mencapai frekuensi motor maksimum adalah sebesar 10 s.

No.	*Parameter List*	Pengaturan Parameter
14	P1121 (*Ramp-down time*)	P1121 merupakan parameter untuk menentukan nilai *ramp–down time. Ramp-down time* adalah waktu yang dibutuhkan oleh motor untuk mengurangi kecepatan motor pada saat motor dalam keadaan frekuensi motor maksimum sampai berhenti. Waktu yang dibutuhkan untuk mencapai frekuensi motor dalam keadaan maksimum sampai berhenti adalah sebesar 10 s.
15	P3900 (*End Quick Commissioning*)	P3900 merupakan parameter untuk menentukan *End Quick Commissioning*. Ada 4 pilihan yaitu : 0 = *No Quick Commissioning (no motor calculation).* *1 = End Quick Commissioning, with factory reset of all other settings* ***(Recommended)*** *2 = End Quick Commissioning, with factory reset of I/O settings.* *3 = End Quick Commissioning, without reset of all other settings.* Setelah semua parameter telah diatur, maka yang perlu dilakukan adalah memilih angka "1", yaitu *End Quick Commissioning* dengan mengatur ulang semua pengaturan sesuai setelan pabrik.

BAB V
PENUTUP

Setelah melakukan perancangan dan pembuatan alat serta pengujian dan analisa, maka dapat ditarik kesimpulan dan saran dari kegiatan yang telah dilakukan untuk pengembangan Tugas Akhir ini.

5.1 Kesimpulan

Dari seluruh tahapan yang sudah dilaksanakan pada penyusunan Tugas Akhir ini, mulai dari studi *literature*, perancangan dan pembuatan sampai pada pengujiannya maka dapat disimpulkan bahwa :

1. Tegangan inputan dengan range 0 sampai 10 Volt dari rangkaian DAC dikonversi ke nilai frekuensi oleh Inverter Sinamics G110 dengan range 0-50 Hz, atau frekuensi naik 5 Hz setiap kenaikan input tegangan 1 Volt.
2. Dengan inputan maksimal Inverter sebesar 10 Volt, kecepatan putar maksimal motor yang dapat dicapai adalah 1600 rpm.
3. Semakin tinggi nilai Kp yang diberikan mengakibatkan kecepatan *steady state* yang dapat dicapai motor juga semakin tinggi dan mendekati *set point*, semakin tinggi nilai Ki yang diberikan mengakibatkan kecepatan waktu *rise time* dan *settling time* juga semakin cepat, sedangkan nilai Kd berpengaruh pada nilai *overshoot.*
4. Nilai setting PID yang didapat dari metode *Trial and Error* untuk Plant Tanpa Beban, Beban I, dan Beban II adalah sebagai berikut :
   Kp : 2,7
   Ki : 1,7
   Kd : 0,1
5. Pembebanan mempengaruhi waktu *rise time* motor, semakin besar pembebanan maka waktu *rise time* akan semakin lama, contohnya tanpa kontrol PID pada *setpoint* 1400 *rise time* yang awalnya 8,4 *detik* mejadi 6,8 *detik* ketika diberi beban I dan naik lagi menjadi 9,6 *detik* ketika diberi beban II. Sedangakan dengan kontrol PID pada *setpoint* 1400 *rise time* yang awalnya 3,1 *detik* mejadi 3,6 *detik* ketika diberi beban I dan naik lagi menjadi 4,9 *detik* ketika diberi beban II

5.2 Saran

Untuk lebih memperbaiki dan menyempurnakan kinerja dari alat ini, maka perlu disarankan :

1. Sebaiknya motor lebih sering dirawat dan dicek kualitasnya agar performanya tetap baik.
2. Karena pengaturan delay sangat mempengaruhi bentuk grafik yang muncul pada *LabVIEW*, disarankan pengaturan *delay* lebih diperhatikan lagi, karena jika penentuan *delay* tidak tepat, maka grafik yang muncul akan tidak sesuai dengan pengukuran kecepatan *real time* sehingga data yang muncul juga menjadi tidak rapi.

DAFTAR PUSTAKA

[1] Arwanjer Semit, "*Perancangan dan Implementasi Kontroller Linear Kuadratik Regulator (LQR) Pada Pengaturan Kecepatan Motor Induksi 3 Phasa*", **Tugas Akhir**, Jurusan Teknik Elektro FTI-ITS, Surabaya, 2014.

[2] Aji Sasmita, "*Perancangan dan Implementasi Direct Torque Control untuk Pengaturan Kecepatan Motor Induksi 3 Fasa Menggunakan Kontroler Fuzzy PI",* **Tugas Akhir**, Jurusan Teknik Elektro FTI-ITS, Surabaya, 2011.

[3] A.Baharsyah,Dimas. **"***Mengatur Kecepatan Motor 3 Phasa Berbeban Rem Elektromagnetik"*, **Tugas Akhir***,* D3 Teknik Elektro FTI-ITS, Surabaya, 2014.

[4] Ogata, Katsuhiko. **Teknik Kontrol Automatik (Sistem Pengaturan) Jilid 1**, Penerbit Erlangga, Jakarta, 1991.

[5] ....., ***Datasheet Inverter SINAMICS G110****,* G110 Operating Instructions Edition 04/2005, SIEMENS, 2005.

[6] ....., ***Datasheet Ethernet Board WIZ110SR,*** WIZ110SR User's Manual (Version1.0), WIZnet, 2007.

[7] ....., ***Datasheet Rotary Encoder****,* Rotary Encoder (Incremental Type) ENA / E50S8 / ENC Series Manual, Autonics, 2009.

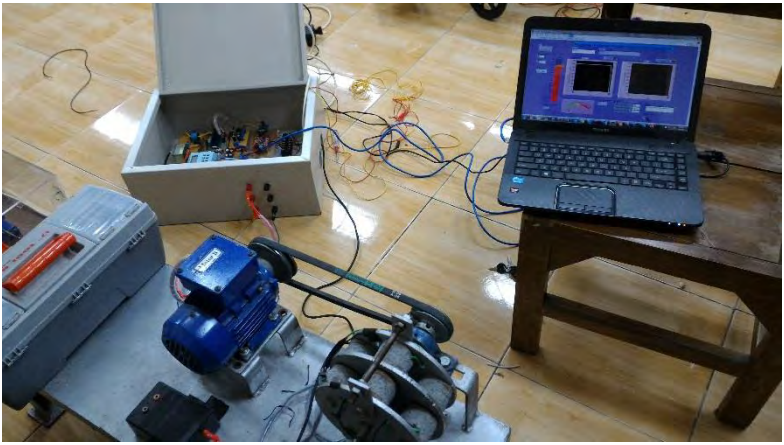
[8] Rafiudin, Rahmat. **Sistem Komunikasi Data Mutakhir Menggunakan ATMega16,** Andi Yogyakarta, Yogyakarta, 2006.

[9] Zuhal, Mahfud. **Dasar Tenaga Listrik dan Elektronika Daya***,* Gramedia, Jakarta, 1998.

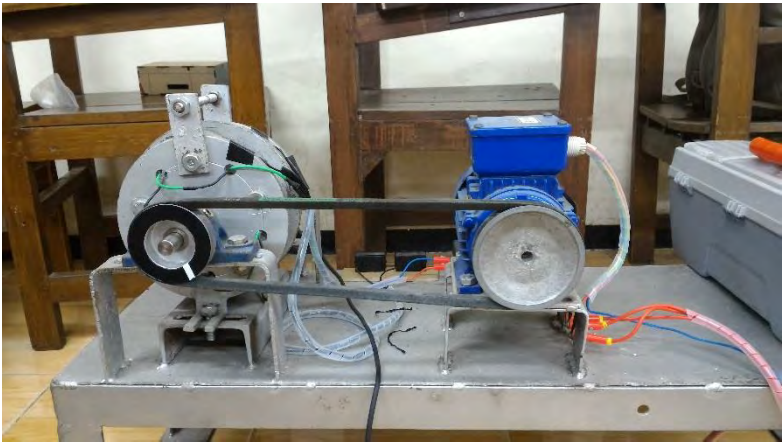
----Halaman ini Sengaja Dikosongkan----

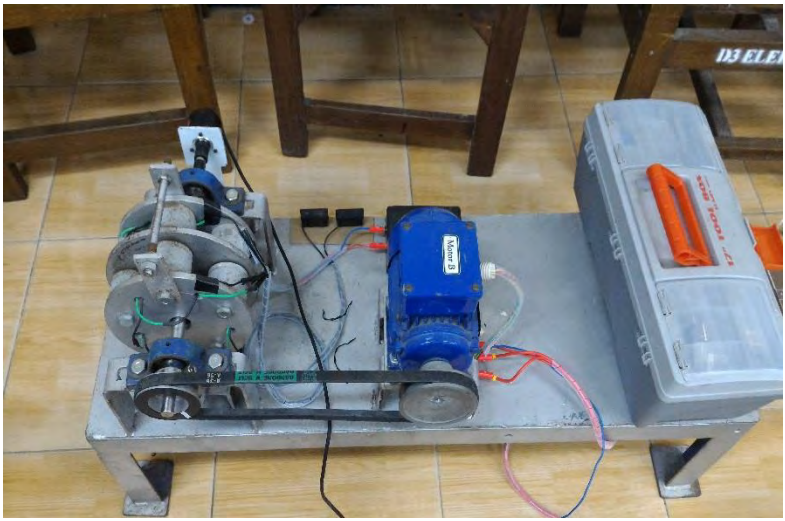
LAMPIRAN A
FOTO

Gambar Sistem Alat Keseluruhan

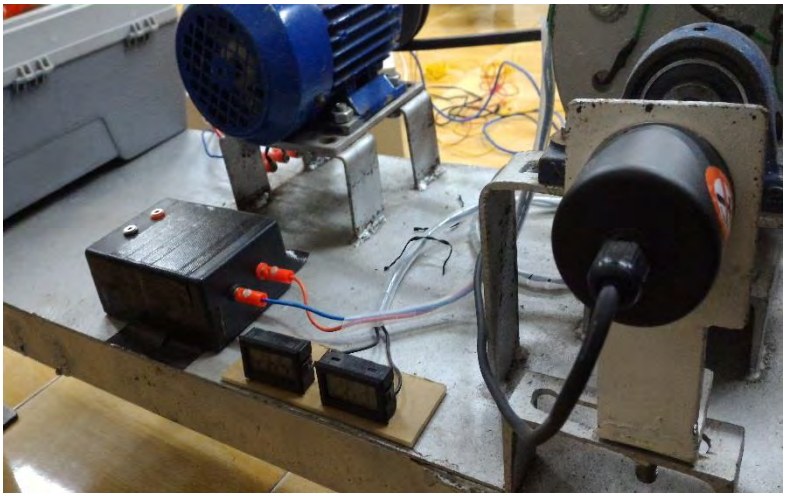


Gambar Motor Dikopel dengan Beban Pengereman





Gambar Peletakkan Sensor Temperatur dan Sensor Kecepatan



Gambar Peletakkan Komponen Pada *Panel Box*

-----Halaman ini sengaja dikosongkan-----

LAMPIRAN B
PROGRAM

B.1 Pemrograman Keseluruhan

```
/*****************************************************
This program was produced by the
CodeWizardAVR V2.05.3 Standard
Automatic Program Generator
© Copyright 1998-2011 Pavel Haiduc, HP InfoTech s.r.l.
http://www.hpinfotech.com

Project :
Version :
Date : 16/05/2016
Author : Cahyo
Company : pngping corp
Comments:


Chip type : ATmega16
Program type : Application
AVR Core Clock frequency : 11,059200 MHz
Memory model : Small
External RAM size : 0
Data Stack size : 256
*****************************************************/

int frekuensi = 0;
float pulsa;
char temp[8];

#include <string.h>
#include <mega16.h>
#include <stdlib.h>
#include <delay.h>
#include <stdio.h>

// Alphanumeric LCD functions
```

```
#include <alcd.h>
char x;
char gabung[20];
// External Interrupt 0 service routine
interrupt [EXT_INT0] void ext_int0_isr(void)
{
// Place your code here
   frekuensi++;
}

#ifndef RXB8
#define RXB8 1
#endif

#ifndef TXB8
#define TXB8 0
#endif

#ifndef UPE
#define UPE 2
#endif

#ifndef DOR
#define DOR 3
#endif

#ifndef FE
#define FE 4
#endif

#ifndef UDRE
#define UDRE 5
#endif

#ifndef RXC
#define RXC 7
#endif

#define FRAMING_ERROR (1<<FE)
```

```
#define PARITY_ERROR (1<<UPE)
#define DATA_OVERRUN (1<<DOR)
#define DATA_REGISTER_EMPTY (1<<UDRE)
#define RX_COMPLETE (1<<RXC)

// USART Receiver buffer
#define RX_BUFFER_SIZE 8
char rx_buffer[RX_BUFFER_SIZE];

#if RX_BUFFER_SIZE <= 256
unsigned char rx_wr_index,rx_rd_index,rx_counter;
#else
unsigned int rx_wr_index,rx_rd_index,rx_counter;
#endif

// This flag is set on USART Receiver buffer overflow
bit rx_buffer_overflow;

// USART Receiver interrupt service routine
interrupt [USART_RXC] void usart_rx_isr(void)
{
char status,data;
status=UCSRA;
data=UDR;
if ((status & (FRAMING_ERROR | PARITY_ERROR |
DATA_OVERRUN))==0)
   {
   rx_buffer[rx_wr_index++]=data;
#if RX_BUFFER_SIZE == 256
   // special case for receiver buffer size=256
   if (++rx_counter == 0) rx_buffer_overflow=1;
#else
   if (rx_wr_index == RX_BUFFER_SIZE) rx_wr_index=0;
   if (++rx_counter == RX_BUFFER_SIZE)
      {
      rx_counter=0;
      rx_buffer_overflow=1;
      }
#endif
```

```
   }
}

#ifndef _DEBUG_TERMINAL_IO_
// Get a character from the USART Receiver buffer
#define _ALTERNATE_GETCHAR_
#pragma used+
char getchar(void)
{
char data;
while (rx_counter==0);
data=rx_buffer[rx_rd_index++];
#if RX_BUFFER_SIZE != 256
if (rx_rd_index == RX_BUFFER_SIZE) rx_rd_index=0;
#endif
#asm("cli")
--rx_counter;
#asm("sei")
return data;
}
#pragma used-
#endif

// USART Transmitter buffer
#define TX_BUFFER_SIZE 8
char tx_buffer[TX_BUFFER_SIZE];

#if TX_BUFFER_SIZE <= 256
unsigned char tx_wr_index,tx_rd_index,tx_counter;
#else
unsigned int tx_wr_index,tx_rd_index,tx_counter;
#endif

// USART Transmitter interrupt service routine
interrupt [USART_TXC] void usart_tx_isr(void)
{
if (tx_counter)
   {
   --tx_counter;
```

```
  UDR=tx_buffer[tx_rd_index++];
#if TX_BUFFER_SIZE != 256
  if (tx_rd_index == TX_BUFFER_SIZE) tx_rd_index=0;
#endif
  }
}

#ifndef _DEBUG_TERMINAL_IO_
// Write a character to the USART Transmitter buffer
#define _ALTERNATE_PUTCHAR_
#pragma used+
void putchar(char c)
{
while (tx_counter == TX_BUFFER_SIZE);
#asm("cli")
if (tx_counter || ((UCSRA & DATA_REGISTER_EMPTY)==0))
  {
  tx_buffer[tx_wr_index++]=c;
#if TX_BUFFER_SIZE != 256
  if (tx_wr_index == TX_BUFFER_SIZE) tx_wr_index=0;
#endif
  ++tx_counter;
  }
else
  UDR=c;
#asm("sei")
}
#pragma used-
#endif

// Standard Input/Output functions
#include <stdio.h>

float Arus, Data_Arus;
char lcd_char[16];

#define ADC_VREF_TYPE 0x60

// Read the 8 most significant bits
```

```
// of the AD conversion result
unsigned char read_adc(unsigned char adc_input)
{
ADMUX=adc_input | (ADC_VREF_TYPE & 0xff);
// Delay needed for the stabilization of the ADC input voltage
delay_us(10);
// Start the AD conversion
ADCSRA|=0x40;
// Wait for the AD conversion to complete
while ((ADCSRA & 0x10)==0);
ADCSRA|=0x10;
return ADCH;
}

// Timer1 overflow interrupt service routine
interrupt [TIM1_OVF] void timer1_ovf_isr(void)
{
// Reinitialize Timer1 value
TCNT1H=0xD5CD >> 8;
TCNT1L=0xD5CD & 0xff;
// Place your code here
pulsa=(float) frekuensi*55/100;
frekuensi = 0;


}
void tampilkan_LCD()
{
   gabung[0]='\0';

   lcd_gotoxy(0,0);
   lcd_putsf("RPM");
   ftoa(pulsa,1,temp);
   lcd_gotoxy(5,0);
   lcd_puts(temp);
   Data_Arus = read_adc(0);
   Arus = ((Data_Arus*0.064811715) - 2.116171548);
   if(Data_Arus <= 35)
   {
```

```
    Arus = 0;
   }

   lcd_gotoxy(0,1);
   lcd_putsf("Arus= ");
   ftoa(Arus,2,lcd_char);
   lcd_puts(lcd_char);
   strcat(gabung,temp);
   strcat(gabung,"&");
   strcat(gabung,lcd_char);
   //delay_ms(70);
   //lcd_clear();
}

// Declare your global variables here

void main(void)
{
// Declare your local variables here


// Input/Output Ports initialization
// Port A initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTA=0x00;
DDRA=0x00;

// Port B initialization
// Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out
Func2=Out Func1=Out Func0=Out
// State7=0 State6=0 State5=0 State4=0 State3=0 State2=0 State1=0
State0=0
PORTB=0x00;
DDRB=0xFF;

// Port C initialization
```

```
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTC=0x00;
DDRC=0x00;

// Port D initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTD=0x00;
DDRD=0x00;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=0xFF
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: 10,800 kHz
// Mode: Normal top=0xFFFF
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer1 Overflow Interrupt: On
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x05;
TCNT1H=0xD5;
```

```
TCNT1L=0xCD;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer2 Stopped
// Mode: Normal top=0xFF
// OC2 output: Disconnected
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;

// External Interrupt(s) initialization
// INT0: On
// INT0 Mode: Falling Edge
// INT1: Off
// INT2: Off
GICR|=0x40;
MCUCR=0x02;
MCUCSR=0x00;
GIFR=0x40;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x04;

// USART initialization
// Communication Parameters: 8 Data, 1 Stop, No Parity
// USART Receiver: On
// USART Transmitter: On
// USART Mode: Asynchronous
// USART Baud Rate: 9600
UCSRA=0x00;
UCSRB=0xD8;
```

```
UCSRC=0x86;
UBRRH=0x00;
UBRRL=0x47;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80;
SFIOR=0x00;

// ADC initialization
// ADC Clock frequency: 691,200 kHz
// ADC Voltage Reference: AREF pin
// ADC Auto Trigger Source: Free Running
ADMUX=ADC_VREF_TYPE & 0xff;
ADCSRA=0xA4;
SFIOR&=0x1F;

// SPI initialization
// SPI disabled
SPCR=0x00;

// TWI initialization
// TWI disabled
TWCR=0x00;

// Alphanumeric LCD initialization
// Connections are specified in the
// Project|Configure|C Compiler|Libraries|Alphanumeric LCD menu:
// RS - PORTC Bit 0
// RD - PORTC Bit 1
// EN - PORTC Bit 2
// D4 - PORTC Bit 4
// D5 - PORTC Bit 5
// D6 - PORTC Bit 6
// D7 - PORTC Bit 7
// Characters/line: 16
lcd_init(16);
```

```
// Global enable interrupts
#asm("sei")

while (1)
      {
      // Place your code here
      tampilkan_LCD();
      puts(gabung);
      x = getchar();
      if (x==0){pulsa=0;};
      PORTB = x;
      delay_ms(250);
      lcd_clear();
      }
}
```

B.2 Pemrograman Pengecekan Tegangan Mikrokontroler *High Voltage*

```
/*****************************************************
This program was produced by the
CodeWizardAVR V2.05.3 Standard
Automatic Program Generator
© Copyright 1998-2011 Pavel Haiduc, HP InfoTech s.r.l.
http://www.hpinfotech.com

Project :
Version :
Date    : 25/05/2016
Author  : Cahyo
Company : pngping corp
Comments:


Chip type               : ATmega16
Program type            : Application
AVR Core Clock frequency: 12,000000 MHz
Memory model            : Small
External RAM size       : 0
Data Stack size         : 256
*****************************************************/

#include <mega16.h>
#include <string.h>
#include <stdlib.h>
#include <delay.h>
#include <stdio.h>
float data;
char lcd_char[16];
// Alphanumeric LCD functions
#include <alcd.h>

#define ADC_VREF_TYPE 0x00

// Read the AD conversion result
```

```
unsigned int read_adc(unsigned char adc_input)
{
ADMUX=adc_input | (ADC_VREF_TYPE & 0xff);
// Delay needed for the stabilization of the ADC input voltage
delay_us(10);
// Start the AD conversion
ADCSRA|=0x40;
// Wait for the AD conversion to complete
while ((ADCSRA & 0x10)==0);
ADCSRA|=0x10;
return ADCW;
}

// Declare your global variables here
void tampilkan_LCD()
{
data = read_adc(0);
//data = read_adc(1);
//data = read_adc(2);
//data = read_adc(3);
//data = read_adc(4);
//data = read_adc(5);
//data = read_adc(6);
//data = read_adc(7);
//adc = (data*1023);
lcd_gotoxy(0,1);
lcd_putsf("data= ");
ftoa(data,2,lcd_char);
lcd_puts(lcd_char);
delay_ms(100);
lcd_clear();
}
void main(void)
{
// Declare your local variables here

// Input/Output Ports initialization
// Port A initialization
```

```
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTA=0x00;
DDRA=0x00;

// Port B initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTB=0xFF;
DDRB=0x00;

// Port C initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTC=0xFF;
DDRC=0x00;

// Port D initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTD=0x00;
DDRD=0x00;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=0xFF
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;
```

```
// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer1 Stopped
// Mode: Normal top=0xFFFF
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer2 Stopped
// Mode: Normal top=0xFF
// OC2 output: Disconnected
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;

// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: Off
MCUCR=0x00;
```

```
MCUCSR=0x00;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x00;

// USART initialization
// USART disabled
UCSRB=0x00;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80;
SFIOR=0x00;

// ADC initialization
// ADC Clock frequency: 93,750 kHz
// ADC Voltage Reference: AREF pin
// ADC Auto Trigger Source: Free Running
ADMUX=ADC_VREF_TYPE & 0xff;
ADCSRA=0xA7;
SFIOR&=0x1F;

// SPI initialization
// SPI disabled
SPCR=0x00;

// TWI initialization
// TWI disabled
TWCR=0x00;

// Alphanumeric LCD initialization
// Connections are specified in the
// Project|Configure|C Compiler|Libraries|Alphanumeric LCD menu:
// RS - PORTD Bit 0
// RD - PORTD Bit 1
// EN - PORTD Bit 2
// D4 - PORTD Bit 4
// D5 - PORTD Bit 5
```

```
// D6 - PORTD Bit 6
// D7 - PORTD Bit 7
// Characters/line: 16
lcd_init(16);

while (1)
      {
      // Place your code here
       tampilkan_LCD();
       PORTB = 255;
       PORTC = 255;
      }
}
```

B.3 Pemrograman Pengecekan Tegangan Mikrokontroler *Low Voltage*

```
/*****************************************************
This program was produced by the
CodeWizardAVR V2.05.3 Standard
Automatic Program Generator
© Copyright 1998-2011 Pavel Haiduc, HP InfoTech s.r.l.
http://www.hpinfotech.com

Project :
Version :
Date    : 25/05/2016
Author  : Cahyo
Company : pngping corp
Comments:


Chip type               : ATmega16
Program type            : Application
AVR Core Clock frequency: 12,000000 MHz
Memory model            : Small
External RAM size       : 0
Data Stack size         : 256
*****************************************************/

#include <mega16.h>
#include <string.h>
#include <stdlib.h>
#include <delay.h>
#include <stdio.h>
float data;
char lcd_char[16];
// Alphanumeric LCD functions
#include <alcd.h>

#define ADC_VREF_TYPE 0x00

// Read the AD conversion result
```

```
unsigned int read_adc(unsigned char adc_input)
{
ADMUX=adc_input | (ADC_VREF_TYPE & 0xff);
// Delay needed for the stabilization of the ADC input voltage
delay_us(10);
// Start the AD conversion
ADCSRA|=0x40;
// Wait for the AD conversion to complete
while ((ADCSRA & 0x10)==0);
ADCSRA|=0x10;
return ADCW;
}

// Declare your global variables here
void tampilkan_LCD()
{
data = read_adc(0);
//data = read_adc(1);
//data = read_adc(2);
//data = read_adc(3);
//data = read_adc(4);
//data = read_adc(5);
//data = read_adc(6);
//data = read_adc(7);
//adc = (data*1023);
lcd_gotoxy(0,1);
lcd_putsf("data= ");
ftoa(data,2,lcd_char);
lcd_puts(lcd_char);
delay_ms(100);
lcd_clear();
}
void main(void)
{
// Declare your local variables here

// Input/Output Ports initialization
// Port A initialization
```

```
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTA=0x00;
DDRA=0x00;

// Port B initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTB=0xFF;
DDRB=0x00;

// Port C initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTC=0xFF;
DDRC=0x00;

// Port D initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTD=0x00;
DDRD=0x00;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=0xFF
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;
```

```
// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer1 Stopped
// Mode: Normal top=0xFFFF
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer2 Stopped
// Mode: Normal top=0xFF
// OC2 output: Disconnected
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;

// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: Off
MCUCR=0x00;
```

```
MCUCSR=0x00;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x00;

// USART initialization
// USART disabled
UCSRB=0x00;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80;
SFIOR=0x00;

// ADC initialization
// ADC Clock frequency: 93,750 kHz
// ADC Voltage Reference: AREF pin
// ADC Auto Trigger Source: Free Running
ADMUX=ADC_VREF_TYPE & 0xff;
ADCSRA=0xA7;
SFIOR&=0x1F;

// SPI initialization
// SPI disabled
SPCR=0x00;

// TWI initialization
// TWI disabled
TWCR=0x00;

// Alphanumeric LCD initialization
// Connections are specified in the
// Project|Configure|C Compiler|Libraries|Alphanumeric LCD menu:
// RS - PORTD Bit 0
// RD - PORTD Bit 1
// EN - PORTD Bit 2
// D4 - PORTD Bit 4
// D5 - PORTD Bit 5
```

```
// D6 - PORTD Bit 6
// D7 - PORTD Bit 7
// Characters/line: 16
lcd_init(16);

while (1)
      {
      // Place your code here
       tampilkan_LCD();
       PORTB = 0;
       PORTC = 0;
      }
}
```

B.4 Pemrograman Pengiriman Data Melalui *Real Term*

```
/*****************************************************
This program was produced by the
CodeWizardAVR V2.05.3 Standard
Automatic Program Generator
© Copyright 1998-2011 Pavel Haiduc, HP InfoTech s.r.l.
http://www.hpinfotech.com

Project :
Version :
Date    : 08/06/2016
Author  : Cahyo
Company : pngping corp
Comments:


Chip type               : ATmega16
Program type            : Application
AVR Core Clock frequency: 12,000000 MHz
Memory model            : Small
External RAM size       : 0
Data Stack size         : 256
*****************************************************/

#include <mega16.h>
#include <stdio.h>
#include <stdlib.h>
#include <delay.h>
 char i;
 char x [8];
// Alphanumeric LCD functions
#include <alcd.h>

#ifndef RXB8
#define RXB8 1
#endif

#ifndef TXB8
```

```
#define TXB8 0
#endif

#ifndef UPE
#define UPE 2
#endif

#ifndef DOR
#define DOR 3
#endif

#ifndef FE
#define FE 4
#endif

#ifndef UDRE
#define UDRE 5
#endif

#ifndef RXC
#define RXC 7
#endif

#define FRAMING_ERROR (1<<FE)
#define PARITY_ERROR (1<<UPE)
#define DATA_OVERRUN (1<<DOR)
#define DATA_REGISTER_EMPTY (1<<UDRE)
#define RX_COMPLETE (1<<RXC)

// USART Receiver buffer
#define RX_BUFFER_SIZE 8
char rx_buffer[RX_BUFFER_SIZE];

#if RX_BUFFER_SIZE <= 256
unsigned char rx_wr_index,rx_rd_index,rx_counter;
#else
unsigned int rx_wr_index,rx_rd_index,rx_counter;
#endif
```

```
// This flag is set on USART Receiver buffer overflow
bit rx_buffer_overflow;

// USART Receiver interrupt service routine
interrupt [USART_RXC] void usart_rx_isr(void)
{
char status,data;
status=UCSRA;
data=UDR;
if ((status & (FRAMING_ERROR | PARITY_ERROR |
DATA_OVERRUN))==0)
   {
   rx_buffer[rx_wr_index++]=data;
#if RX_BUFFER_SIZE == 256
   // special case for receiver buffer size=256
   if (++rx_counter == 0) rx_buffer_overflow=1;
#else
   if (rx_wr_index == RX_BUFFER_SIZE) rx_wr_index=0;
   if (++rx_counter == RX_BUFFER_SIZE)
      {
      rx_counter=0;
      rx_buffer_overflow=1;
      }
#endif
   }
}

#ifndef _DEBUG_TERMINAL_IO_
// Get a character from the USART Receiver buffer
#define _ALTERNATE_GETCHAR_
#pragma used+
char getchar(void)
{
char data;
while (rx_counter==0);
data=rx_buffer[rx_rd_index++];
#if RX_BUFFER_SIZE != 256
if (rx_rd_index == RX_BUFFER_SIZE) rx_rd_index=0;
#endif
```

```
#asm("cli")
--rx_counter;
#asm("sei")
return data;
}
#pragma used-
#endif

// USART Transmitter buffer
#define TX_BUFFER_SIZE 8
char tx_buffer[TX_BUFFER_SIZE];

#if TX_BUFFER_SIZE <= 256
unsigned char tx_wr_index,tx_rd_index,tx_counter;
#else
unsigned int tx_wr_index,tx_rd_index,tx_counter;
#endif

// USART Transmitter interrupt service routine
interrupt [USART_TXC] void usart_tx_isr(void)
{
if (tx_counter)
   {
   --tx_counter;
   UDR=tx_buffer[tx_rd_index++];
#if TX_BUFFER_SIZE != 256
   if (tx_rd_index == TX_BUFFER_SIZE) tx_rd_index=0;
#endif
   }
}

#ifndef _DEBUG_TERMINAL_IO_
// Write a character to the USART Transmitter buffer
#define _ALTERNATE_PUTCHAR_
#pragma used+
void putchar(char c)
{
while (tx_counter == TX_BUFFER_SIZE);
#asm("cli")
```

```
if (tx_counter || ((UCSRA & DATA_REGISTER_EMPTY)==0))
   {
   tx_buffer[tx_wr_index++]=c;
#if TX_BUFFER_SIZE != 256
   if (tx_wr_index == TX_BUFFER_SIZE) tx_wr_index=0;
#endif
   ++tx_counter;
   }
else
   UDR=c;
#asm("sei")
}
#pragma used-
#endif

// Standard Input/Output functions
#include <stdio.h>

// Declare your global variables here
void hah ()
{
lcd_gotoxy(0,0);
lcd_putsf("128");
}
void main(void)
{
// Declare your local variables here

// Input/Output Ports initialization
// Port A initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTA=0x00;
DDRA=0x00;

// Port B initialization
```

```
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTB=0x00;
DDRB=0x00;

// Port C initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTC=0x00;
DDRC=0x00;

// Port D initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTD=0x00;
DDRD=0x00;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=0xFF
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer1 Stopped
// Mode: Normal top=0xFFFF
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
```

```
// Input Capture on Falling Edge
// Timer1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer2 Stopped
// Mode: Normal top=0xFF
// OC2 output: Disconnected
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;

// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: Off
MCUCR=0x00;
MCUCSR=0x00;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x00;

// USART initialization
// Communication Parameters: 8 Data, 1 Stop, No Parity
// USART Receiver: On
```

```
// USART Transmitter: On
// USART Mode: Asynchronous
// USART Baud Rate: 9600
UCSRA=0x00;
UCSRB=0xD8;
UCSRC=0x86;
UBRRH=0x00;
UBRRL=0x4D;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80;
SFIOR=0x00;

// ADC initialization
// ADC disabled
ADCSRA=0x00;

// SPI initialization
// SPI disabled
SPCR=0x00;

// TWI initialization
// TWI disabled
TWCR=0x00;

// Alphanumeric LCD initialization
// Connections are specified in the
// Project|Configure|C Compiler|Libraries|Alphanumeric LCD menu:
// RS - PORTC Bit 0
// RD - PORTC Bit 1
// EN - PORTC Bit 2
// D4 - PORTC Bit 4
// D5 - PORTC Bit 5
// D6 - PORTC Bit 6
// D7 - PORTC Bit 7
// Characters/line: 16
lcd_init(16);
```

```
// Global enable interrupts
#asm("sei")

while (1)
      {
      // Place your code here
      hah();
      i=0; i=128 ; i++;;
      }
}
```

B.5 Pemrograman Rangkaian *DAC* dan Penguat *Non Inverting*

```
/*****************************************************
This program was produced by the
CodeWizardAVR V2.05.3 Standard
Automatic Program Generator
© Copyright 1998-2011 Pavel Haiduc, HP InfoTech s.r.l.
http://www.hpinfotech.com

Project :
Version :
Date    : 11/04/2016
Author  : Cahyo
Company : pngping corp
Comments:


Chip type               : ATmega16
Program type            : Application
AVR Core Clock frequency: 12,000000 MHz
Memory model            : Small
External RAM size       : 0
Data Stack size         : 256
*****************************************************/

#include <mega16.h>
#include <delay.h>
// Declare your global variables here

void main(void)
{
// Declare your local variables here

// Input/Output Ports initialization
// Port A initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
```

```
PORTA=0x00;
DDRA=0x00;

// Port B initialization
// Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out
Func2=Out Func1=Out Func0=Out
// State7=0 State6=0 State5=0 State4=0 State3=0 State2=0 State1=0
State0=0
PORTB=0x00;
DDRB=0xFF;

// Port C initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTC=0x00;
DDRC=0x00;

// Port D initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTD=0x00;
DDRD=0x00;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=0xFF
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer1 Stopped
```

```
// Mode: Normal top=0xFFFF
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer2 Stopped
// Mode: Normal top=0xFF
// OC2 output: Disconnected
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;

// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: Off
MCUCR=0x00;
MCUCSR=0x00;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x00;
```

```
// USART initialization
// USART disabled
UCSRB=0x00;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80;
SFIOR=0x00;

// ADC initialization
// ADC disabled
ADCSRA=0x00;

// SPI initialization
// SPI disabled
SPCR=0x00;

// TWI initialization
// TWI disabled
TWCR=0x00;

while (1)
      {
      // Place your code here
        PORTB = 0b11111111 ;
      }
}
```

LAMPIRAN C
DATASHEET

1. ***Datasheet*** **Inverter Sinamics G110**

1 Overview

1.1 The SINAMICS G110

The SINAMICS G110s are a range of frequency inverters for controlling the speed of three phase AC motors, incorporating the CPM 110 controlled power module. The various models available range from 120 W to 3.0 kW single-phase input.

The inverters are microprocessor-controlled and use state-of-the-art Insulated Gate Bipolar Transistor (IGBT) technology. This makes them reliable and versatile. A special pulse-width modulation method with selectable pulse frequency permits quiet motor operation. Comprehensive protective functions provide excellent inverter and motor protection.

The SINAMICS G110 CPM110 with its default factory settings is ideal for a large range of simple V/f motor control applications. Using the comprehensive range of programmable parameters provided with the inverter, the unit can be adapted for a wide range of applications. Parameters can be changed using either USS communications or the Basic Operator Panel (BOP).

The SINAMICS G110 is available in two variants; the Analog controlled variant and the USS controlled variant utilizing RS485 protocol. They are available as filtered and unfiltered inverters including a "Flat Plate" version which completes the range. They can be used in both 'stand-alone' applications as well as being integrated into 'Automation Systems".

1.2 Features

Main Characteristics

- Easy installation
- Easy commission
  - Quick commissioning
  - Reset function (allowing the reset of all values to the preset factory defaults)
- Rugged EMC design
- Can be operated on IT line supplies (unfiltered variants)
- 1 digital output – Isolated optocoupler
- 3 digital inputs (non-isolated)
- 1 Analog input, AIN: 0 – 10 V (Analog variant only). Can be used as 4th digital input
- High pulse frequencies for low-noise motor operation
- Status information and alarm messages using the Basic Operator Panel
- Optional Basic Operator Panel with the capability to clone parameter sets
- USS communications interface (USS variant only)
- PC to RS232 Connection Kit available

The inverter can be screened using the methodology shown in Figure 2-8 below.

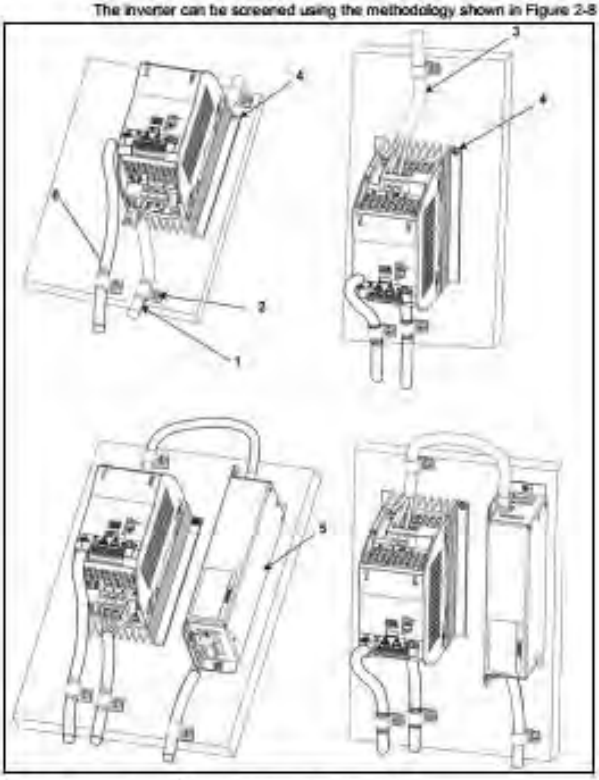


Figure 2-8 Wiring Guidelines to Minimize the Effects of EMI

Legend

1 Motor cable
2 Use suitable clips to fix motor and control cable screens securely to metal back plate
3 Mains power cable
4 Retaining screw (inverter fixing bolts)
5 Line commutating Choke
6 Control cable

2.10 SINAMICS G110 Flat Plate Variant

The SINAMICS G110 Flat Plate variant has been designed to allow greater flexibility in the installation of the inverter by an advanced user as an individual unit or as part of an automation system. Adequate measures must be taken to ensure the correct heat dissipation, which may require an additional external heatsink outside the electrical cabinet enclosure (see Table 2-3 on page 30).

The installation procedures, both mechanical and electrical, starting on page 17 of these Operating Instructions should be performed unless otherwise stated below. Ensuring that all warnings and cautions given throughout the procedures are strictly observed.

Figure 2-9 SINAMICS G110 Flat Plate Variant

WARNING

Operation with the input voltages less than 230 V and 50 Hz or with a pulse frequency greater than 8 kHz will cause an additional heat load on the inverter. These factors must be taken into account when designing the installation conditions and must be verified by a practical load test.

Cooling Considerations

1. For the correct dimensioning of the panel, please consult the panel builder and his technical documentation.
2. For the correct dimensioning of the external heatsink, please refer to the technical data as shown in Table 2-1 on page 20.
3. The rear plate must be able to withstand at least 95°C during fault free operation and carry the heat load (steel or aluminium plate) under the full load conditions and the maximum operating temperatures [-10°C to +50°C (14°F to 122 °F)]. For further information see Table 2-3 on page 30.
4. The minimum clearance distances of 30 mm at both sides and 100 mm vertical spacing at both sides must be observed.
5. It is recommended that the mounding area of the rear plate has to be, as a minimum, at least the same area of the flat plate on the inverter.
6. Side by side or stacked mounting is not allowed for the SINAMICS G110 Flat Plate inverter.

Installation

1. Prepare the mounting surface for the inverter using the dimensions given in Table 2-1 on page 20.
2. Ensure that any rough edges are removed from the drilled holes.
3. Ensure the inverter's Flat Plate is clean and free from dust and grease.
4. The mounting surface for the Flat Plate and if applicable the external heatsink must be:
   - Clean and free from dust and grease.
   - Smooth.

- Free from dents and holes.
- Made of metal (steel or aluminium).
- Must not be painted.
- Must be free from rust.

5. Apply coating of heatsink/thermal contact paste to the inverter's flat plate.
6. Ensure that the contact paste is spread evenly over the rear surface of the Flat Plate.
7. Mount the inverter using four M4 screws.
8. Ensure that the inverter is mounted securely and the M4 screws are lightened to the correct torque of 2.5 Nm (22.12 lbf.in).
9. If required, connect the external heatsink on the other side of the rear plate, ensuring an evenly spread thermal contact paste is apply.
10. When the installation is completed the effectiveness of the cooling should be verified by a load test.
11. Check that there is no trip F0004.

Table 2-3 Flat Plate Power Losses and Thermal Specifications*

	120 W	250 W	370 W	550 W	750 W
Operating temperature range (°C)	-10 to +50				-10 to +40
Total losses (W)	22	28	36	43	54
Linesicle and control losses (W)	9	10	12	13	15
Recommended thermal resistance of heatsink (K/W)	3.0	2.2	1.6	1.2	1.2
Recommended output current (A)	0.9	1.7	2.3	3.2	3.9

*The losses given in Table 2-3 are applicable for units fitted with 25 m of screened cable

3.2 Commission Modes

Basic commissioning of the SINAMICS G110 inverter can be performed by using one of the following methods which are suitable for a large range of applications:

- Using the inverter with its factory default settings by connecting analogue and digital inputs or using the RS485 connections (see Section 3.3.1 on page 35).
- Using the optional Basic Operator Panel (BOP) (see Section 3.3.2. on page 37).

Advanced commissioning allows the user to suitably adapt the inverter to the specific application. Section 3.4 contains information about:

- Using a PLC to communicate with the inverter via the USS protocol (see Section 3.4.1 on page 42).
- Using the PC Tool "STARTER" which communicates with the inverter via USS protocol (see Section 3.4.1 on page 43).
- Optimal configuration of the inverter by setting parameters using 'Quick Commissioning' (see Section 3.4.4 on page 46).
- Resetting inverter parameters to factory default (see Section 3.4.5 on page 49).
- Connecting a PTC temperature sensor to the inverter (see Section 3.4.6 on page 49).
- Parameter Cloning with the BOP (see Section 3.4.7 on page 49).

Notes

When utilizing USS communications, a common 0 V reference connection is required between all devices on the USS bus. Terminal 10 on the control board can be used for this purpose.

The SINAMICS G110 inverter is available as two variants:

1. **Analog Variant**
   The analog variant is suited to stand-alone applications. This variant is designed to be controlled using external switches and a potentiometer utilizing the analog input and digital inputs. Switches and potentiometers are not provided as standard with the inverter.
2. **USS Variant**
   The USS variant is suited to inverter networks. This variant is designed to be controlled utilizing the USS protocol via the RS485 communications interface. A number of inverters can then be connected and controlled via the same communications bus.

The variant can be identified by reading the MLFB from the rating label and comparing it to the MLFBs listed in Table 7-4 and Table 7-5 on pages 68 and 69 respectively.

Since the SINAMICS G110 is available in two variants; different options for the commissioning of the inverters are available to the user. These commissioning options are described in the following sections covering the 'Basic Commissioning' and 'Advanced Commissioning'.

3.3 Basic Commissioning

The SINAMICS G110 is supplied with default parameter settings to cover the following basic operation:

- The motor rating data; voltage, current and frequency data has already been keyed into the inverter to ensure that the motor is compatible with the inverter. (A Siemens standard motor is recommended).
- Linear V/f motor speed, controlled by an analogue potentiometer, or via the RS485 connection using the USS variant.
- Maximum speed 3000 min^{-1} corresponding to a 2-pole motor with 50 Hz (3600 min^{-1} with 60 Hz); controllable using a potentiometer via the inverter's analogue input, or via the RS485 connection using the USS variant.
- Ramp-up time / Ramp-down time = 10 s.

Changing the motor base frequency

The default motor base frequency of the SINAMICS G110 inverter is 50 Hz. In some parts of the world motors are designed for a base frequency of 60 Hz. Changing the motor default base frequency is accomplished using the DIP switch which is provided on the front of the inverter to select the required base frequency.

A small screwdriver will be required to change the position of the DIP switches.

DIP switch 1 is used to change the motor base frequency of the inverter, by default it is in the position '50 Hz'. See Figure 3-2. In the default position (50 Hz) the output power will be displayed in kW (if a BOP is fitted to the inverter). In addition, all motor related parameters will be calculated using the 50 Hz setting.

To change the motor base frequency to 60 Hz, DIP switch 1 must be set to the position '60 Hz'.

The DIP switch must be set to the required frequency before power is applied to the inverter. On power up the inverter will read the DIP switch setting and calculate the following motor related parameters:

- Rated Motor Frequency (P0310)
- Maximum Motor Frequency (P1082)
- Reference Frequency (P2000)

Figure 3-2 Motor Base Frequency DIP Switch and Bus Termination

3.3.1 Factory Settings

The inverter has already been programmed at the factory for standard V/f applications on a Siemens standard four-pole 3-phase induction motor, that has the same power rating as the inverters.

Controlling the speed of the motor is accomplished by connecting the analog inputs on the analog variant (switches and the potentiometer are not supplied with the inverter) or via the RS485 connections on the USS variant as shown in Figure 3-3 below.

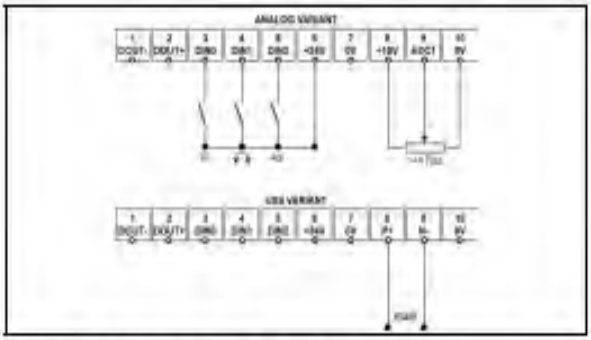


Figure 3-3 Basic operation – Analog and USS Variants

The inverter can be used with its default settings for a wide range of applications. The default settings are shown in Table 3-1 (Analog variant) and Table 3-2 (USS variant). The terminal layout is shown in Figure 3-3 above.

Note

The motor base frequency might have to be changed as described in the previous section on page 34.

Table 3-1 Factory settings for operation using the standard inverter (Analog Variant)

Description	Terminals	Parameter Default	Default Operation
Frequency Setpoint Source	9	P1000 = 2	Analog Input
Command Source	3,4 & 5	P0700 = 2	(see below)
Digital Input 0	3	P0701 = 1	ON/OFF1
Digital Input 1	4	P0702 = 12	Reverse
Digital Input 2	5	P0703 = 9	Fault Acknowledge
Control Method	-	P0727=0	Siemens Standard Control

With the default settings of the inverter (Analog variant) the following is possible:

- Start and stopping the motor (DIN0 via external switch)
- Reversing the motor (DIN1 via external switch)
- Fault Acknowledgement (DIN2 via external switch).

Bus termination on USS variant

The USS variant of the SINAMICS G110 inverter uses RS485 protocols to communicate with the controlling system and all other inverters connected to the network.

It is necessary to terminate the last inverter on the network bus. This is achieved by setting the Bus Termination DIP switches on the front of the inverter to the 'Bus Termination' position (as shown in Figure 3-2 on page 34). It is important that both DIP switches (2 and 3) are set to the 'Bus Termination' position (not in the OFF position). A small screwdriver will be required to change the position of the DIP switches.

3.3.2 Commissioning with the Optional Basic Operator Panel

If the optional 'Basic Operator Panel' (BOP) is available, the control signals and speed reference can easily be set by pressing the relevant buttons. The BOP also provides easy access to the inverter parameters. This section describes how to commission and start the inverter with the minimum of effort, using the BOP.

For advanced use of the BOP to perform for example, the full inverter commissioning see Section 3.4.1 on page 42 or refer to Section 3.4.7 on page 49 for information about parameter cloning using the BOP.

For instructions on how to fit the BOP to the inverter, please refer to Appendix C on page 82 and for a description of the buttons see Appendix D on page 83.

- The BOP must be connected directly to the inverter and not remotely connected via a cable.
- The BOP can be fitted to and removed from the inverter whilst power is applied.
- The inverter will automatically recognize that the BOP has been fitted to the inverter and give the user access to the parameters. To run the inverter (start/stop, setpoint) using the BOP, parameters P0700 (command source, i.e. start/stop, reverse, jog) and P1000 (frequency setpoint) have to be set to 1. Alternatively, P0719 can be set to 11 which is described below.

Figure 3-4 BOP

Note

The motor base frequency might have to be changed as described in the previous section on page 34.

Changing parameters with the Basic Operator Panel

The description below serves as an example that shows how to change any parameters using the BOP. These examples can also be used as a guide to configuring the inverter for running via the BOP (start/stop commands and the frequency setpoint are input on the BOP).

Changing P0003 – parameter access level

	Step	Result on display
1	Press P to access parameters	r0000
2	Press ▲ until P0003 is displayed	P0003
3	Press P to display the parameter value	1
4	Press ▲ or ▼ to set the required value (set to 3)	3
5	Press P to confirm and store the value	P0003
6	All level 1 to level 3 parameters are now visible to the user.	

Changing P0719 an indexed parameter – setting BOP control

	Step	Result on display
1	Press P to access parameters	r0000
2	Press ▲ until P0719 is displayed	P0719
3	Press P to access the parameter value	in000
4	Press ▲ or ▼ to select index 1	in001
5	Press P to display actual set value	0
6	Press ▲ or ▼ to the required value	11
7	Press P to confirm and store the value	P0719
8	Press ▼ until r0000 is displayed	r0000
9	Press P to return the display to the standard drive display (as defined by the customer)	

Figure 3-5 Changing parameters via the BOP

NOTE

In some cases - when changing parameter values - the display on the BOP shows BUSY. This means the inverter is busy with tasks of a higher priority.

Changing single digits in parameter values

For changing the parameter value rapidly, the single digits of the display can be changed by performing the following actions:

1. Ensure you are in the parameter value changing level (see section "Changing parameters with Basic Operator Panel" above).
2. Press Fn (function button), which causes the right hand digit to blink.
3. Change the value of this digit by pressing ▲ or ▼.
4. Press Fn (function button) again causes the next digit to blink.
5. Perform steps 2 to 4 until the required value is displayed.
6. Press the P to leave the parameter value changing level.

NOTE

The function button Fn may also be used to acknowledge a fault condition.

Commissioning of motorpoti (MOP) function

Simple motor speed control can be achieved using the motorpoti (MOP) function of the optional BOP (for using the MOP see also P1031 and P1040 in the Parameter List).

The BOP motor control functions are disabled by default. To control the motor via the BOP, the following settings must be completed (see also "Changing parameters with the Basic Operator Panel" above):

➢ P0719 = 11 (enables start/stop button on the BOP and enables the motor potentiometer setpoint from the BOP).

Alternatively set:

- P0700 = 1 (enables the start/stop button on the BOP).
- P1000 = 1 (this enables the motor potentiometer setpoints).

1. Press the I button to start the motor.
2. Press the ▲ button while the motor is turning. Motor speed increases to 50 Hz.
3. When the inverter reaches 50 Hz, press button ▼. Motor speed and displayed value are decreased.
4. Change the direction of rotation by pressing the ↔ button.
5. Stop the motor by pressing the O button.

If the BOP has been set as the command source (P0700 = 1 or P0719 = 10 - 15), the inverter will stop if the BOP is removed.

3.4.4 Quick Commissioning (P0010=1)

Quick commissioning is an easy way to optimally configure the SINAMICS G110 inverter to a specific motor. The motor data, taken from the motor rating label, is entered into the inverter and then the inverter calculates the dependent control and protection parameters.

An alternative to quick commissioning is parameter cloning (see page 49) that can be used if a large number of inverters are to be commissioned to the same specific motor.

NOTE

It is only possible to change motor parameters when quick commissioning is enabled (P0010=1).

It is important that parameter P0010 is used for commissioning and P0003 is used to select the number of parameters to be accessed. The P0003 parameter allows a group of parameters to be selected that will enable quick commissioning. Parameters such as Motor settings and Ramp settings are included.

At the end of the quick commissioning sequence, P3900 should be selected, which, when set to 1, will carry out the necessary motor calculations and clear all other parameters (not included in P0010=1) to the default settings. This will only happen in the Quick Commissioning mode.

Parameter P0010 is automatically reset to the value 0 when P3900 > 0. The inverter can only be run if P0010 has been set back to 0.

NOTE

We recommend commissioning is performed according to this scheme. Nevertheless an expert user is allowed to perform the commissioning with the filter functions of P0004.

Motor data for parameterization

Figure 3-6 below indicates where to find the relevant motor data on the motor rating plate. Figure 3-6 is for illustration purposes only and the values within this figure should not be keyed into your inverter, rather the values from your own motor's rating plate should be keyed into the inverter.

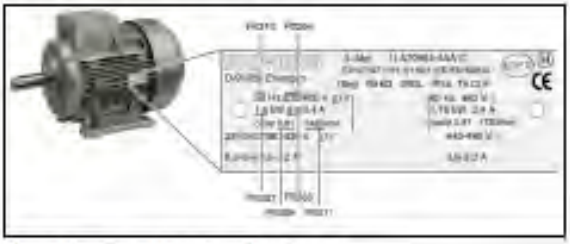
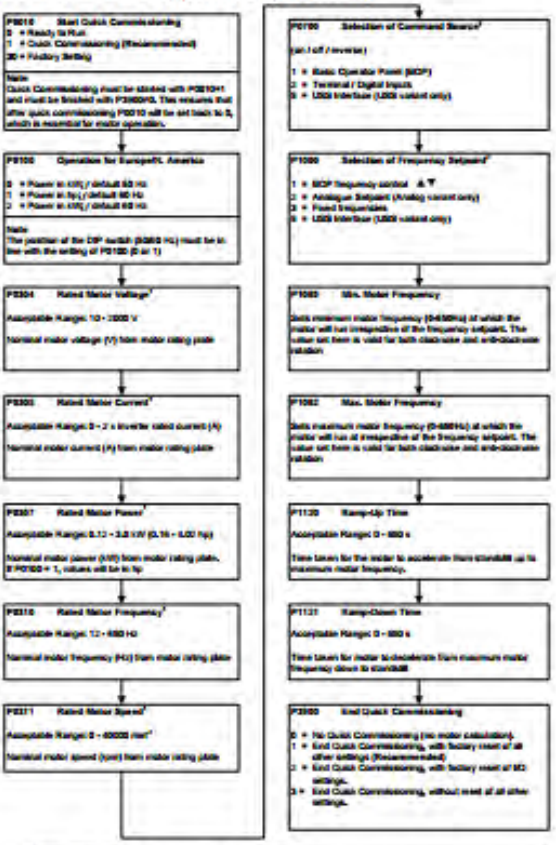


Figure 3-6 Typical Motor Rating Plate Example

Flow chart Quick Commissioning (Level 1 Only – P0003=1)



1. Motor related parameters - please refer to the motor rating plate.
2. Indicates parameters that contain more detailed lists of possible settings for use in specific applications. Please refer to the Parameter List.

D Description of the BOP

Panel/Button	Function	Effects
r0000	Indicates Status	The LCD displays the settings currently used by the converter.
I	Start motor	Pressing the button starts the inverter. This button is disabled by default. To enable this button set P0700 = 1.
O	Stop motor	OFF1 Pressing the button causes the inverter to come to a standstill at the selected ramp down rate. OFF2 Pressing the button twice (or once long) causes the motor to coast to a standstill. This function is always enabled.
	Change direction	Press this button to change the direction of rotation of the motor. Reverse is indicated by a minus (-) sign or a flashing decimal point. Disabled by default, to enable set P0700 = 1.
JOG	Jog motor	Pressing this button while the inverter has no ON command causes the motor to start and run at the preset jog frequency. The inverter stops when the button is released. Pressing this button when the inverter/motor is running has no effect.
FN	Functions	This button can be used to view additional information. Pressing and holding the button for 2 seconds from any parameter during operation, shows the following: 1. DC link voltage (indicated by d – units V). 2. Output voltage (indicated by o – units V). 3. Output frequency (Hz). 4. The value selected in P0005. Additional presses will toggle around the above displays. A short press of the button will acknowledge a fault condition. **Jump Function** From any parameter (rXXXX or PXXXX) a short press of the Fn button will immediately jump to r0000, you can then change another parameter, if required. Upon returning to r0000, pressing the Fn button will return you to your starting point.
P	Access parameters	Pressing this button allows access to the parameters.
▲	Increase value	Pressing this button increases the displayed value.
▼	Decrease value	Pressing this button decreases the displayed value.

2. *Datasheet* Ethernet Board WIZ110SR

1.2 Specification

Items	Description
MCU	8051 Compliant (having internal 62K Flash, 16K SRAM, 2K EEPROM)
TCP/IP	W5100 (Ethernet PHY Embedded)
Network Interface	10/100 Mbps auto-sensing RJ-45 Connector
Serial Interface	RS232
Serial Signal	TXD, RXD, RTS, CTS, GND
Serial Parameters	Parity : None, Even, Odd
	Data Bits : 7,8
	Flow Control : None, RTS/CTS, XON/XOFF
	Speed : up to 230Kbps
Input Voltage	DC 5V
Power Consumption	Under 180mA
Temperature	0°C ~ 80°C (Operation), -40°C ~ 85°C (Storage)
Humidity	10 ~ 90%

Table 1. WIZ110SR Specification

1.3 Products Contents

	WIZ110SR Board
	CD (Option / including Firmware, Configuration Tool Program Manual and other related materials)

5. Hardware Description of WIZ110SR

5.1 Parameters

- Power 5V DC / 180mA
- Dimension 75 x 50 x 17 (L x W x H)
- Temperature Operating : 0 ~ 80 ℃
- Ethernet 10/100 Base-T Ethernet (Auto detection)
- Serial interface RS-232

5.2 Specification

- MCU 8051 Compliant
- FLASH 62KByte (MCU Internal)
- SRAM 16KByte (MCU Internal)
- EEPROM 2KByte (MCU Internal)

5.3 Board Dimensions and Pin Assignment

5.3.1 Dimensions

2. WIZ110SR Board

2.1 Block Diagram

Figure 1. Block Diagram

WIZ110SR is a protocol converter that transmits the data sent by serial equipment as TCP/IP data type and converts back the TCP/IP data received through the network into serial data to transmit back to the equipment. When the data is received from serial port, it is sent to W5100 by MCU. If any data is transmitted from Ethernet, it is received in the internal buffer if W5100, and sent to the serial port by MCU. MCU in the module controls the data according to the configuration value that user defined.

<TCP server mode>

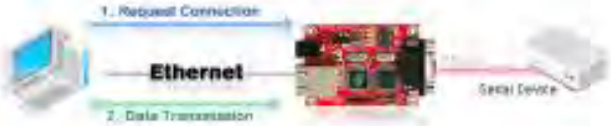


Figure 3. TCP Server mode

At the TCP Server mode, WIZ110SR waits for the connection requests.
TCP Server mode can be useful when the monitoring center tries to connect to the device (where WIZ110SR is installed) in order to check the status or provide the commands. In normal time WIZ110SR is on the waiting status, and if there is any connection request from the monitoring center, data communication is processed and connection is closed.

In order to operate this mode, Local IP, Subnet, Gateway Address and Local Port Number should be configured first.

As illustrated in the above figure, data transmission proceeds as follows,
1. The host connects to the WIZ1q0SR which is configured as TCP Server mode.
2. As the connection is established, data can be transmitted in both directions - from the host to the WIZ110SR, and from the WIZ110SR to the host

<TCP client mode>

Figure 4. TCP Client mode

If WIZ110SR is set as TCP Client, it tries to establish connection to the server.
To operate this mode, Local IP, Subnet, Gateway Address, Server IP, and Server port number should be set. If server IP had domain name, use DNS function.

In TCP Client mode, WIZ110SR can actively establish a TCP connection to a host computer when power is supplied.

As illustrated in the above figure, data transmission proceeds as follows:
1. As power is supplied, WIZ110SR board operating as TCP client mode actively establishes a connection to the server.
2. If the connection is complete, data can be transmitted in both directions - from the host to the WIZ110SR and from WIZ110SR to the host

<Mixed mode>
In this mode, WIZ110SR normally operates as TCP Server and waits for the connection request from the peer. However, if WIZ110SR receives data from the serial device before connection is established, it changes to the client mode and sends the data to the server IP. Therefore, at the mixed mode, the server mode is operated prior to the client mode.

As like TCP Server mode, the Mixed mode is useful for the case that the monitoring center tries to connect to the serial device (in which WIZ110SR is used) to check device status. In addition to this, if any emergency occurs in the serial device, the module will change to Client mode to establish the connection to the server and deliver the emergency status of the device

❺ Use UDP mode

Figure 5. UDP mode

At the UDP mode, the connection establishment is not defined. Just set the IP address and port number of the peer and send the data.

4. Demonstration and Test

This chapter will show how WIZ110SR operates through demonstration and test. The hardware and software for the testing is shown in below table.

	PC	WIZ110SR
Hardware	1) RS232 Port 2) LAN Port	1) WIZ110SR Board 2) Serial Cable 3) LAN Cable 4) DC5V Power Adaptor
Software	1) Configuration Tool Program 2) Hyper Terminal	

4.1 Hardware Interface

Figure 12. WIZ110SR Interface

Install the board as below steps.

STEP 1: Connect the WIZ110SR to the network by using RJ45 Ethernet cable.

STEP 2: Connect the WIZ110SR to the serial device by using Serial cable

STEP 3: Connect the power adaptor to WIZ110SR board.

3. ***Datasheet* Sensor *Rotary Encoder***

E50S Series

Diameter ø50mm Shaft type Incremental Rotary encoder

Line-up

Features

- 12-24VDC power supply of line driver output(Line-up)
- Suitable for measuring angle, position, revolution, speed, acceleration and distance
- Power supply : 5VDC, 12-24VDC ±5%

Applications

- Various tooling machinery, packing machine and general industrial machinery etc.

Please read "Caution for your safety" in operation manual before using.

Ordering information (Former name : ENB)

E50S - 8 - 5000 - 3 - N - 24 - [ ]

Series	Shaft diameter	Pulse/1Revolution	Output phase	Output	Power supply	Cable
Diameter ø50mm, shaft type	ø8mm	Refer to resolution	2: A, B 3: A, B, Z 4: A, $\overline{A}$, B, $\overline{B}$ 6: A, $\overline{A}$, B, $\overline{B}$, Z, $\overline{Z}$	T: Totem pole output N: NPN open collector output V: Voltage output L: Line driver output	5: 5VDC ±5% 24: 12-24VDC ±5%	No mark: Without type T: Cable outgoing connector type(-) C: Cable side outgoing connector integrated type CS: Cable outgoing connector integrated type

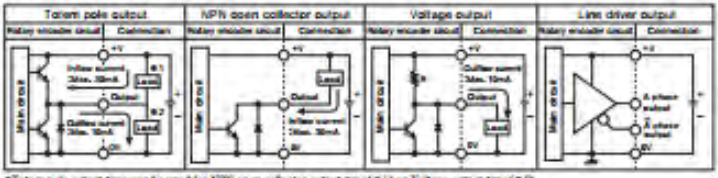
- Standard : E50S8-5000-3-N-24
- Cable length : 2m

Specifications

Item			Diameter ø50mm shaft type of incremental rotary encoder
Resolution(P/R)			(Note1) 1, 2, 5, 10, 12, 15, 20, 30, 36, 50, 60, 75, 100, 120, 125, 150, 180, 200, 240, 250, 254, 300, 360, 400, 500, 600, 720, 800, 900, 1000, 1024, 1200, 1250, 1440, 2000, 2048, 2500, 3000, 3600, 4000, 5000, 6000
Electrical specification	Output phase		A, B, Z phase(Line driver : A, $\overline{A}$, B, $\overline{B}$, Z, $\overline{Z}$ phase)
	Phase difference of output		Phase difference between A and B : $\frac{T}{4} \pm \frac{T}{8}$ (T=1cycle of A phase)
	Control output	Totem pole output	• Low : Load current:Max. 30mA, Residual voltage : Max. 0.4VDC • High : Load current:Max. 10mA, Output voltage(Power supply 5VDC):Min. (Power supply-2.0)VDC, Output voltage(Power supply 12-24VDC):Min. (Power supply-3.0)VDC
		NPN open collector output	Load current : Max. 30mA, Residual voltage : Max. 0.4VDC
		Voltage output	Load current : Max. 10mA, Residual voltage : Max. 0.4VDC
		Line driver output	• Low : Load current : Max. 20mA, Residual : Max. 0.5VDC • High : Load current : Max. -20mA, Output voltage(Power supply 5VDC) : Min. 2.5VDC, Output voltage(Power voltage 12-24VDC) : Min. (Power supply-3.0)VDC
	Response time (Rise/Fall)	Totem pole output / NPN open collector output / Voltage output / Line driver output	Max. 1μs(Cable length : 2m, (tank = 20mA))
	Max. response frequency		300kHz
	Power supply		• 5VDC ±5%(Ripple P-P : Max. 5%) • 12-24VDC ±5%(Ripple P-P : Max. 5%)
	Current consumption		Max. 80mA (disconnection of the load), Line driver output : Max. 50mA (disconnection of the load)
	Insulation resistance		Min. 100MΩ (at 500VDC megger between all terminals and case)
	Dielectric strength		750VAC 50/60Hz for 1 minute (between all terminals and case)
	Connection		Cable outgoing type, 2000mm cable outgoing connector type, Connector integrated type(Note6, Note7)
Mechanical specification	Starting torque		Max. 10gf · cm(0.007N · m)(Note2) / Max. 300gf · cm(0.029N · m)(Note3)
	Moment of inertia		Max. 30g · cm^2(5 × 10^{-6}kg · m^2)(Note2) / Max. 300g · cm^2(5.1 × 10^{-5}kg · m^2)(Note3)
	Shaft loading		Radial : 10kgf, Thrust : 2.5kgf
	Max. allowable revolution	(Note4)	5000rpm
Vibration			1.5mm amplitude at frequency of 10 to 55Hz(for 1 min.) in each of X, Y, Z directions for 2 hours
Shock			Max. 75G
Ambient temperature			-10 to 70°C(at non-freezing status), Storage : -25 to 85°C
Ambient humidity			35 to 85%RH, Storage : 35 to 90%RH
Protection			Normal type, Cable outgoing connector type : IP50(IEC standard)(Note5), Connector integrated type : IP40(IEC standard)
Cable			ø5mm, 5P, Length : 2m, Shield cable(Line driver output : ø6mm, 8P) (AWG 20, Core wire diameter : 0.09mm, No. of core wire : 20, Insulator out diameter : ø1.1mm)
Accessory			ø8mm coupling, bracket
Approval			Normal type : CE (Except for line driver output)
Unit weight			Approx. 270g, Connector integrated type : 190g

- (Note1) "-" pulse is only for A, B phase(Line drive output is for A, $\overline{A}$, B, $\overline{B}$ phase).
- (Note2) This value is for normal type, cable outgoing connector type (Protection: IP50).
- (Note3) This value is for normal type, cable outgoing connector type (Protection: IP50)(connector integrated type (Protection: IP40)).
- (Note4) Make sure that max. response revolution should be lower than or equal to max. allowable revolution when selecting the resolution.
  (Max. response revolution(rpm)) = $\frac{\text{Max. response frequency}}{\text{Resolution}} \times 60\text{ sec.}$
- (Note5) "Normal type, cable outgoing connector type is quotet as IP54 protection.

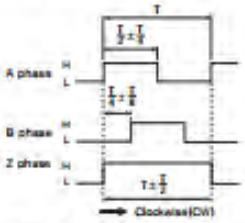
■Control output diagram



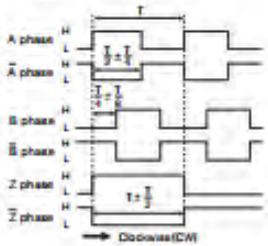
※Totem pole output type can be used for NPN open collector output type(※1) or Voltage output type(※2).
※The output circuit of A, B, Z phase are the same, Line driver output is A, Ā, B, B̄, Z, Z̄.

■Output waveform

※Totem pole output / NPN open collector output / Voltage output



※Line driver output

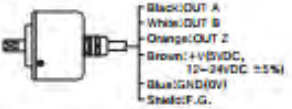


※CW : Right turn as from the shaft

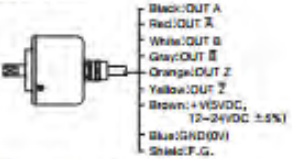
■Connections

■Normal type

※Totem pole output / NPN open collector output / Voltage output



※Line driver output



※Unused wires must be insulated.
※The shield cable and metal case of encoder must be grounded(F.G.)

■Cable outgoing connector/ Connector integrated type

※Totem pole output / NPN open collector output / Voltage output

※Line driver output

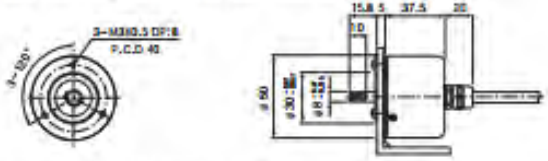
Totem pole output / NPN open collector output / Voltage output			Line driver output		
Pin No	Function	Cable color	Pin No	Function	Cable color
①	OUT A	Black	①	OUT A	Black
②	OUT B	White	②	OUT Ā	Red
③	OUT Z	Orange	③	+V	Brown
④	+V	Brown	④	GND	Blue
⑤	GND	Blue	⑤	OUT B	White
⑥	F.G.	Shield	⑥	OUT B̄	Gray
			⑦	OUT Z	Orange
			⑧	OUT Z̄	Yellow
			⑨	F.G.	Shield

※F.G.(Field Ground) : It must be grounded separately.

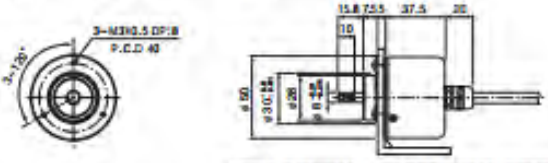
●Dimensions

(Unit:mm)

■Normal type, Cable outgoing connector type(Protection : IP50)



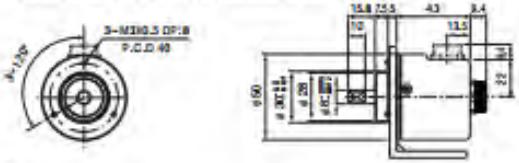
■Normal type, Cable outgoing connector type(Protection : IP64)



Cable for normal type	Cable for Cable outgoing connector type
ø 5mm, SP:Line driver output(5P), Length:3000mm, Shield cable	ø 5mm, SP:Line driver output(5P), Length:350mm, Shield cable

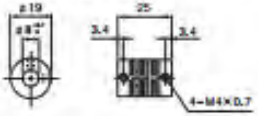
※ Connector cable is sold separately and see C-6 for specifications.

■Rear/Side connector integrated type(Protection: IP65)



●Bracket

●Coupling(E50S)



- Parallel misalignment: Max. 0.2mm
- Angular misalignment: Max. 3°
- End-play: Max. 0.2mm

※For parallel misalignment, angular misalignment, End-play terms, refer to F-69 page.

※For Encoder coupling(E50) detail information, refer to F-68 page.

※ may result in damage to this unit.

■ Outline

This unit is very useful to control length, angle and position by converting revolution value of shaft into number of pulse as an optical incremental Encoder.

■ Ordering information

ENA	- 5000	- 2	- N	- 24
Series	Pulse/1Revolution	Output phase	Output	Power supply
Shaft type to be mounted at the side (Shaft diameter ø10mm)	See resolution	2: A, B 3: A, B, Z	T: Totem Pole output N: NPN open collector output V: Voltage output	5: 5VDC±5% 24: 12~24VDC ±5%

※Standard: ENA-PULSE-2-N-24 ※Standard: A, B

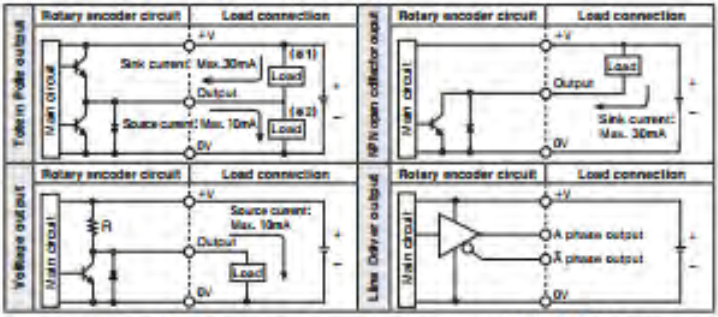
E50S	8	- 8000	- 3	- N	- 24	-
Series	Shaft diameter	Pulse/1Revolution	Output phase	Output	Power supply	Cable
Diameter ϕ50mm shaft type	ø8mm	See resolution	2: A, B 3: A, B, Z 4: A, $\overline{A}$, B, $\overline{B}$ 6: A, $\overline{A}$, B, $\overline{B}$, Z, $\overline{Z}$	T: Totem Pole output N: NPN open collect output V: Voltage output L: Line Driver ouput	5: 5VDC±5% 24: 12~24VDC ±5%	No mark: Normal type (※)C: Cable outgoing connector type

※Standard: E50S8-PULSE-3-N-24 ※Cable length:250mm

ENC	- 1	- 1	- N	- 24	-
Series	Output phase	Min. measuring unit	Output	Power supply	Cable
Wheel type	1: A, B	1: 1mm 2: 1cm 3: 1m 4: 0.01yd 5: 0.1yd 6: 1yd	T: Totem Pole output N: NPN open collector output V: Voltage output	5: 5VDC±5% 24: 12~24VDC ±5%	No mark: Normal type (※)C: Cable outgoing connector type

※Cable length:250mm

■ Control output diagram



※ The output circuit of A, B, Z phase are the same.(Line Driver output is A, $\overline{A}$, B, $\overline{B}$, Z, $\overline{Z}$ phase)
※ Totem Pole output can be used for NPN open collector type(※1) or voltage output type(※2).

※ The above specification are subject to change without notice.

4. ***Datasheet* IC MAX232**

TEXAS INSTRUMENTS

MAX232, MAX232I

SLLS047M – FEBRUARY 1989 – REVISED NOVEMBER 2014

MAX232x Dual EIA-232 Drivers/Receivers

1 Features

- Meets or Exceeds TIA/EIA-232-F and ITU Recommendation V.28
- Operates From a Single 5-V Power Supply With 1.0-μF Charge-Pump Capacitors
- Operates up to 120 kbit/s
- Two Drivers and Two Receivers
- ±30-V Input Levels
- Low Supply Current: 8 mA Typical
- ESD Protection Exceeds JESD 22
  - 2000-V Human-Body Model (A114-A)
- Upgrade With Improved ESD (15-kV HBM) and 0.1-μF Charge-Pump Capacitors is Available With the MAX202 Device

2 Applications

- TIA/EIA-232-F
- Battery-Powered Systems
- Terminals
- Modems
- Computers

3 Description

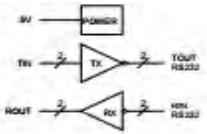
The MAX232 device is a dual driver/receiver that includes a capacitive voltage generator to supply TIA/EIA-232-F voltage levels from a single 5-V supply. Each receiver converts TIA/EIA-232-F inputs to 5-V TTL/CMOS levels. These receivers have a typical threshold of 1.3 V, a typical hysteresis of 0.5 V, and can accept ±30-V inputs. Each driver converts TTL/CMOS input levels into TIA/EIA-232-F levels.

Device Information[1]

ORDER NUMBER	PACKAGE (PIN)	BODY SIZE
MAX232x	SOIC (16)	9.90 mm × 3.91 mm
	SOIC (16)	10.30 mm × 7.50 mm
	PDIP (16)	19.30 mm × 6.35 mm
	SOP (16)	10.3 mm × 5.30 mm

(1) For all available packages, see the orderable addendum at the end of the datasheet.

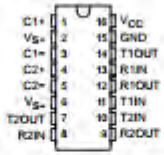
4 Simplified Schematic



6 Pin Configuration and Functions

Top View

MAX232 . . . D, DW, N, OR NS PACKAGE
MAX232I . . . D, DW, OR N PACKAGE
(TOP VIEW)



Pin Functions

PIN NAME	PIN NO.	TYPE	DESCRIPTION
C1+	1	—	Positive lead of C1 capacitor
VS+	2	O	Positive charge pump output for storage capacitor only
C1−	3	—	Negative lead of C1 capacitor
C2+	4	—	Positive lead of C2 capacitor
C2−	5	—	Negative lead of C2 capacitor
VS−	6	O	Negative charge pump output for storage capacitor only
T2OUT, T1OUT	7, 14	O	RS232 line data output (to remote RS232 system)
R2IN, R1IN	8, 13	I	RS232 line data input (from remote RS232 system)
R2OUT, R1OUT	9, 12	O	Logic data output (to UART)
T2IN, T1IN	10, 11	I	Logic data input (from UART)
GND	15	—	Ground
VCC	16	—	Supply Voltage, Connect to external 5V power supply

7 Specifications

7.1 Absolute Maximum Ratings(1)

over operating free-air temperature range (unless otherwise noted)

			MIN	MAX	UNIT
V_{CC}	Input Supply voltage range(2)		−0.3	6	V
V_+	Positive output supply voltage range		V_{CC} − 0.3	15	V
V_-	Negative output supply voltage range		−0.3	−15	V
V_I	Input voltage range	T1IN, T2IN	−0.3	V_{CC} + 0.3	V
		R1IN, R2IN		±30	
V_O	Output voltage range	T1OUT, T2OUT	V_- − 0.3	V_+ + 0.3	V
		R1OUT, R2OUT	−0.3	V_{CC} + 0.3	
	Short-circuit duration	T1OUT, T2OUT	Unlimited		
T_J	Operating virtual junction temperature			150	°C

(1) Stresses beyond those listed under *Absolute Maximum Ratings* may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated under *Recommended Operating Conditions* is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.
(2) All voltages are with respect to network GND.

7.2 Handling Ratings

			MIN	MAX	UNIT
T_{stg}	Storage temperature range		−65	150	°C
$V_{(ESD)}$	Electrostatic discharge	Human body model (HBM), per ANSI/ESDA/JEDEC JS-001, all pins(1)	0	2000	V
		Charged device model (CDM), per JEDEC specification JESD22-C101, all pins(2)	0	1000	

(1) JEDEC document JEP155 states that 500-V HBM allows safe manufacturing with a standard ESD control process.
(2) JEDEC document JEP157 states that 250-V CDM allows safe manufacturing with a standard ESD control process.

7.3 Recommended Operating Conditions

			MIN	NOM	MAX	UNIT
V_{CC}	Supply voltage		4.5	5	5.5	V
V_{IH}	High-level input voltage (T1IN,T2IN)		2			V
V_{IL}	Low-level input voltage (T1IN, T2IN)				0.8	V
R1IN, R2IN	Receiver input voltage				±30	V
T_A	Operating free-air temperature	MAX232	0		70	°C
		MAX232I	−40		85	

7.4 Thermal Information

THERMAL METRIC(1)		MAX232xD	MAX232xDW	MAX232xN	MAX232xNS	UNIT
		SOIC	SOIC wide	PDIP	SOP	
		16 PINS	16 PINS	16 PINS	16 PINS	
$R_{\theta JA}$	Junction-to-ambient thermal resistance	73	57	67	64	°C/W

(1) For more information about traditional and new thermal metrics, see the *IC Package Thermal Metrics* application report (SPRA953).

7.5 Electrical Characteristics — Device

over recommended ranges of supply voltage and operating free-air temperature (unless otherwise noted) (see Figure 6)

PARAMETER	TEST CONDITIONS(1)	MIN	TYP(2)	MAX	UNIT
I_{CC}	Supply current	V_{CC} = 5.5V, all outputs open, T_A = 25°C		8	10	mA

(1) Test conditions are C1–C4 = 1 µF at V_{CC} = 5 V ± 0.5 V.
(2) All typical values are at V_{CC} = 5 V, and T_A = 25°C.

7.6 Electrical Characteristics — Driver

over recommended ranges of supply voltage and operating free-air temperature (unless otherwise noted)

PARAMETER			TEST CONDITIONS[1]	MIN	TYP[2]	MAX	UNIT
V_{OH}	High-level output voltage	T1OUT, T2OUT	R_L = 3 kΩ to GND	5	7		V
V_{OL}	Low-level output voltage[3]	T1OUT, T2OUT	R_L = 3 kΩ to GND		−7	−5	V
r_o	Output resistance	T1OUT, T2OUT	V_{S+} = V_{S-} = 0, V_O = ±2 V	300			Ω
I_{OS}[4]	Short-circuit output current	T1OUT, T2OUT	V_{CC} = 5.5 V, V_O = 0 V		±10		mA
I_{IS}	Short-circuit input current	T1IN, T2IN	V_I = 0			200	µA

(1) Test conditions are C1–C4 = 1 µF at V_{CC} = 5 V ± 0.5 V.
(2) All typical values are at V_{CC} = 5 V, T_A = 25°C.
(3) The algebraic convention, in which the least-positive (most negative) value is designated minimum, is used in this data sheet for logic voltage levels only.
(4) Not more than one output should be shorted at a time.

7.7 Electrical Characteristics — Receiver

over recommended ranges of supply voltage and operating free-air temperature (unless otherwise noted)

PARAMETER			TEST CONDITIONS[1]	MIN	TYP[2]	MAX	UNIT
V_{OH}	High-level output voltage	R1OUT, R2OUT	I_{OH} = −1 mA	3.5			V
V_{OL}	Low-level output voltage[3]	R1OUT, R2OUT	I_{OL} = 3.2 mA			0.4	V
V_{IT+}	Receiver positive-going input threshold voltage	R1IN, R2IN	V_{CC} = 5 V, T_A = 25°C		1.7	2.4	V
V_{IT-}	Receiver negative-going input threshold voltage	R1IN, R2IN	V_{CC} = 5 V, T_A = 25°C	0.8	1.2		V
V_{hys}	Input hysteresis voltage	R1IN, R2IN	V_{CC} = 5 V	0.2	0.5	1	V
r_i	Receiver input resistance	R1IN, R2IN	V_{CC} = 5 V, T_A = 25°C	3	5	7	kΩ

(1) Test conditions are C1–C4 = 1 µF at V_{CC} = 5 V ± 0.5 V.
(2) All typical values are at V_{CC} = 5 V, T_A = 25°C.
(3) The algebraic convention, in which the least-positive (most negative) value is designated minimum, is used in this data sheet for logic voltage levels only.

7.8 Switching Characteristics

over recommended ranges of supply voltage and operating free-air temperature (unless otherwise noted)

PARAMETER		TEST CONDITIONS[1]	MIN	TYP[2]	MAX	UNIT
SR	Driver slew rate	R_L = 3 kΩ to 7 kΩ, see Figure 4			30	V/µs
SR(t)	Driver transition region slew rate	see Figure 5		3		V/µs
	Data rate	One TOUT switching		120		kbit/s
$t_{PLH(R)}$	Receiver propagation delay time, low- to high-level output	TTL load, see Figure 3		500		ns
$t_{PHL(R)}$	Receiver propagation delay time, high- to low-level output	TTL load, see Figure 3		500		ns

(1) Test conditions are C1–C4 = 1 µF at V_{CC} = 5 V ± 0.5 V.

LAMPIRAN D
INISIALISASI INVERTER

Pengaturan Parameter *Quick Comissioning Inverter Sinamics G110*

No.	*Parameter List*	Pengaturan Parameter
1	P0010 (*Start Quick Commissioning*)	P0010 merupakan parameter pengaturan awal untuk memulai mengubah parameter dari *inverter*. Dengan cara menekan tombol "P" yang ada di *inverter*, maka untuk memulai *Quick Comissioning* dengan memilih "1", untuk menyimpan parameter yang sudah diatur dengan cara menekan tombol "P" pada *inverter*.
2	P003 (*User Access Level*)	P003 merupakan parameter untuk menyetujui penggunaan *Quick Comissioning* jadi parameter ini merupakan parameter pertama yang muncul dalam proses *Quick Comissioning*. Untuk mengakses parameter maka dengan menekan tombol "P", kemudian akan muncul 3 pilihan yang terdapat dalam pengaturan ini. Memilih "1", kemudian menekan kembali tombol "P" untuk menyimpan nilai parameter.
3	P0100 (*Operation for Europe / America*)	P0100 merupakan parameter untuk memilih frekuensi operasi yang akan digunakan untuk menggerakkan motor. Ada 3 pilihan dalam pengaturan ini: *0= Power in kW; f default 50 Hz* *1= Power in hp; f default 60 Hz* *2 = Power in kW; f default 60 Hz* Karena frekuensi yang ada di Indonesia sendiri hanya menyediakan jaringan listrik dengan frekuensi 50Hz sama seperti pada negara-negara Eropa, sehingga diharuskan memilih mode *Europe* yaitu "0"

No.	*Parameter List*	Pengaturan Parameter
4	P0304 (*Rated Motor Voltage*)	P0304 merupakan parameter untuk menentukan nilai suplai tegangan motor, pada bagian ini kisaran yang diperbolehkan yaitu 10 –2000 V. Dalam mengisi parameter ini, nilai yang dimasukkan harus sesuai dengan informasi yang ada pada *name plate* motor. Besar nominal tegangan motor (*Volt*) yang tertera pada *name plate* adalah 380V, sehingga nilai yang harus diisi adalah 380
5	P0305 (*Rated Motor Current*)	P0305 merupakan parameter untuk nilai arus nominal dari motor. Pada bagian ini kisaran yang diperbolehkan yaitu 0 – 2x. Nilai Arus nominal yang ada pada *name plate* motor yaitu 0,62 A, sehingga nilai yang harus diisi adalah 0,62
6	P0307 (*Rated Motor Power*)	P0307 merupakan parameter yang menentukan nilai daya motor. Pada bagian ini kisaran yang diperbolehkan adalah 0,12 – 3,0 *KW* (0,16– 4,02 *HP*). Karenna pada Parameter P0100 kita mengisikan 1 maka nominal daya diisikan dalam bentuk HP, besar nominal daya motor yang kita gunakan yaitu 0,25 HP, sehingga nilai yang harus diisi adalah 0,25
7	P0310 (*Rated Motor Frequency*)	P0310 merupakan parameter yang menentukan nilai frekuensi motor. Pada bagian ini kisaran yang diperbolehkan 12 – 650 Hz. Besar nominal frekuensi motor yang tertera pada *name plate* yaitu 50 Hz.
8	P0311 (*Rated Motor Speed*)	P0311 merupakan pengaturan untuk menentukan nilai kecepatan motor, pada bagian ini kisaran yang diperbolehkan 0 –40000 rpm. Besar nominal kecepatan motor (*rpm*) pada *name plate* yaitu 1310 *rpm*.

No.	*Parameter List*	Pengaturan Parameter
9	P0700 (*Selection of Command Source*)	P0700 merupakan parameter untuk pemilihan sumber perintah, dimana nantinya akan muncul tiga pilihan : *1 = Basic Operator Panel (BOP)* *2 = Terminal / Digital Inputs* *5 = USS Interface (USS variant only)* karena semua pengaturan berasal dari *inverter* itu sendiri tanpa memerlukan perangkat lain, maka memilih angka "1"*Basic Operator Panel*.
10	P1000 (*Selection of Frequency Setpoint*)	P1000 merupakan parameter untuk menentukan pengontrolan frekuensi *inverter*. Ada 4 pilihan ketika kita akan menentukan metode pengontrolan frekuensi pada *inverter* yaitu : *1 = BOP frequency control* *2 = Analogue Setpoint (Analog variant only)* *3 = Fixed frequencies* *5 = USS Interface (USS variant only)* Karena untuk mengendalikan motor tiga fasa menggunakan Mikrokontroler *ATMega16* dimana Mikrokontroler ini memberikan tegangan kerja 0–10 V, maka untuk pengendalian frekuensinya menggunakan pilihan "2", yaitu *analog set point*.
11	P1080 (*Minimum Frequency*)	P1080 merupakan parameter untuk menentukan nilai minimal frekuensi motor dengan kisaran frekuensi 0–650 Hz. Motor yang digunakan diatur minimal frekuensi motor sebesar 0 Hz.
12	P1082 (*Maximum Frequency*)	P1082 merupakan parameter untuk menentukan nilai maksimum frekuensi motor dengan kisaran sebesar 0–650 Hz, dimana motor yang digunakan frekuensinya diatur maksimal sebesar 50 Hz.
13	P1120 (*Ramp-up Time*)	P1120 merupakan parameter untuk menentukan nilai *ramp-up time. Ramp-up time* adalah waktu yang dibutuhkan oleh motor dari keadaan diam sampai frekuensi motor maksimum. Waktu yang dibutuhkan untuk mencapai frekuensi motor maksimum adalah sebesar 10 s.

No.	*Parameter List*	Pengaturan Parameter
14	P1121 (*Ramp-down time*)	P1121 merupakan parameter untuk menentukan nilai *ramp–down time. Ramp-down time* adalah waktu yang dibutuhkan oleh motor untuk mengurangi kecepatan motor pada saat motor dalam keadaan frekuensi motor maksimum sampai berhenti. Waktu yang dibutuhkan untuk mencapai frekuensi motor dalam keadaan maksimum sampai berhenti adalah sebesar 10 s.
15	P3900 (*End Quick Commissioning*)	P3900 merupakan parameter untuk menentukan *End Quick Commissioning*. Ada 4 pilihan yaitu : 0 = *No Quick Commissioning (no motor calculation).* *1 = End Quick Commissioning, with factory reset of all other settings* ***(Recommended)*** *2 = End Quick Commissioning, with factory reset of I/O settings.* *3 = End Quick Commissioning, without reset of all other settings.* Setelah semua parameter telah diatur, maka yang perlu dilakukan adalah memilih angka "1", yaitu *End Quick Commissioning* dengan mengatur ulang semua pengaturan sesuai setelan pabrik.

DAFTAR RIWAYAT HIDUP

Nama	:	Ika Hasfritasari
TTL	:	Sidoarjo, 24 Juni 1995
Jenis Kelamin	:	Perempuan
Agama	:	Islam
Alamat	:	Jalan Demang Sari, RT 02/ RW 01 Desa Keboan Anom, Kec. Gedangan, Kab. Sidoarjo
Telp/HP	:	087855624929
E-mail	:	hasfritasari.ika@gmail.com

RIWAYAT PENDIDIKAN

1. 2001 – 2007 : SD Negeri Keboan Anom
2. 2007 – 2010 : SMP Negeri 1 Gedangan
3. 2010 – 2013 : SMA Muhammadiyah 2 Sidoarjo
4. 2013 – 2016 : D3 Teknik Elektro, Program Studi Teknik Elektro Industri - FTI Institut Teknologi Sepuluh Nopember (ITS)

PENGALAMAN KERJA

1. Kerja Praktek di PT.ALTER TRADE INDONESIA (ATINA)
2. Kerja Praktek di PT. PLN (Persero) Rayon Sidoarjo Kota

PENGALAMAN ORGANISASI

1. Staff Divisi Big Event Periode 2014/2015 HIMAD3TEKTRO, FTI - ITS

-----Halaman ini sengaja dikosongkan-----

DAFTAR RIWAYAT HIDUP

Nama	:	Cahyo Aditya Laksono
TTL	:	Sidoarjo, 21 September 1994
Jenis Kelamin	:	Laki - Laki
Agama	:	Islam
Alamat	:	Pondok Sidokare Indah W-13
Telp/HP	:	089675833977
E-mail	:	cahyoadityaa@gmail.com

RIWAYAT PENDIDIKAN

1. 2001 – 2007 : SD Negeri Sidokare 2
2. 2007 – 2010 : SMP Negeri 2 Buduran
3. 2010 – 2013 : SMA Negeri 4 Sidoarjo
4. 2013 – 2016 : D3 Teknik Elektro, Program Studi Teknik Elektro Industri - FTI Institut Teknologi Sepuluh Nopember (ITS)

PENGALAMAN KERJA

1. Kerja Praktek di PT.ALTER TRADE INDONESIA (ATINA)
2. Kerja Praktek di PT. PLN (Persero) Rayon Sidoarjo Kota

PENGALAMAN ORGANISASI

-